

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号
特開2001-147800
(P2001-147800A)

(43)公開日 平成13年 5月29日 (2001. 5. 29)

(51)Int.Cl. ⁷	識別記号	F I	テーマコード [*] (参考)
G 0 6 F 7/24		G 0 6 F 7/24	A
12/00	5 6 0	12/00	5 6 0 F
12/06	5 3 0	12/06	5 3 0 A
15/16	6 2 0	15/16	6 2 0 A
17/30		15/40	3 1 0 B
審査請求 未請求 請求項の数14 O L (全 34 頁) 最終頁に続く			

(21)出願番号 特願平11-330997

(22)出願日 平成11年11月22日 (1999. 11. 22)

(71)出願人 599074648

ターボデータラボラトリー有限会社

東京都台東区松が谷 1-9-12 S P K ビ
ルディング604号

(72)発明者 古庄 晋二

神奈川県横浜市神奈川区松見町 4丁目1101
番地 7 コートハウス菊名804号

(74)代理人 100103632

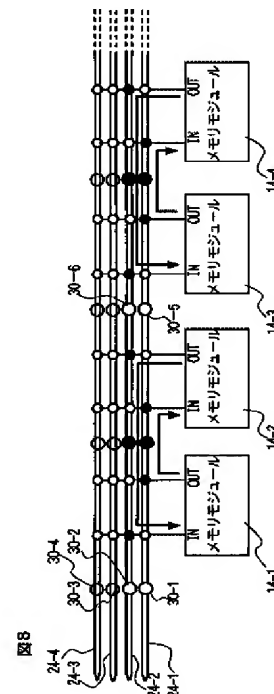
弁理士 窪田 英一郎 (外 1 名)

(54)【発明の名称】 情報処理システム、並びに、この情報処理システムを利用したソート方法、コンパイル方法およびジョイン方法

(57)【要約】

【課題】 著しく高速に、かつ、安定した処理時間で、配列のソートなどを実現する。

【解決手段】 分散メモリ型の情報処理装置において、提示メモリモジュール14-1、14-3が自己のメモリモジュール内でソートされた要素を順位番号とともに、スイッチ30などにより分割されたバス24を介して、判定メモリモジュール14-2、14-4に与える。判定メモリモジュールは、与えられた順位番号に基づき、受理した要素の順位番号の候補を示す仮想順位番号を算出して、当該仮想順位番号を、他のバス24を介して、提示メモリモジュールに返送する。提示メモリモジュールは、前記仮想順位番号を受理した場合に、当該仮想順位番号にしたがって、要素の順位番号を更新する。



【特許請求の範囲】

【請求項1】 CPUモジュールと、それぞれがMPUおよびRAMコアとを有する複数のメモリモジュールと、前記CPUとメモリモジュールとの接続、および／または、任意のメモリモジュール間の接続をなす複数組のバスとを備え、前記一以上のメモリモジュールにより把握される配列に関する処理が、CPUから各メモリモジュールのMPUに与えられるインストラクションにより、各メモリモジュールのMPUの作動により実行されるように構成された分散メモリ型情報処理システムであって、

前記メモリモジュールのMPUが、自己の把握する配列の部分構成する要素のソートを実行して、前記要素を特定の順序にしたがって並べ替えるソート手段と、前記自己の把握する前記部分が配列中に占める位置にしたがって、前記ソートされた要素を、その順位番号とともに、他のメモリモジュールに所定のバスを介して伝達し、或いは、所定のバスを介して、他のメモリモジュールからの前記要素および順位番号を受理するI/Oと、前記要素および順位番号を受理した場合に、自己の把握する要素との比較により、受理した要素の順位番号の候補である仮想順位番号を算出して、前記他のメモリモジュールに返送する順位番号算出手段と、前記仮想順位番号を受理した場合に、当該仮想順位番号にしたがって、要素の順位を確定する順位確定手段とを備え、

前記要素および順位番号を送出する側のメモリモジュールである提示メモリモジュールと、前記要素および順位番号受理して仮想順位番号を算出する側のメモリモジュールである判定メモリモジュールとの通信により、前記配列の要素の順位番号を確定することを特徴とする情報処理システム。

【請求項2】 前記メモリモジュールが、確定した順位番号にしたがって、処理対象となる要素を特定して何れかのバスに送出する要素特定／送出手段と、

前回の処理対象となった要素と送出された要素とを比較する要素比較手段と、

同一の要素が送出された場合には、その値をカウントアップする、同一の要素の存在数を示す同一値個数カウンタとを備え、前記要素比較手段が、前回の処理対象となった要素と送出された要素とが異なると判断した場合に、前回の処理対象となった要素、および、当該要素に関する同一値個数カウンタの値を関連付けて何れかの送出するように構成され、さらに、何れかのメモリモジュールが、

送出された前回の処理対象となった要素および関連するカウンタの値を受理して、これらを関連付け、かつ、受理した順序で配置する配列を備えたことを特徴とする請求項1に記載の情報処理システム。

【請求項3】 前記メモリモジュールが、

前記要素比較手段が、前回の処理対象となった要素と送出された要素とが異なると判断した場合に、その値をカウントアップする、重複のない順位番号を示す値番号カウンタと、

送出された要素に関して、前回の処理対象となった要素と送出された要素とが同一の場合には値番号カウンタの値を、重複のない当該要素の順位番号と決定し、その一方、これらが異なる場合には、カウントアップされた値番号カウンタの値を、重複のない当該要素の順位番号と決定して、当該順位番号を更新する順位番号更新手段とを備えたことを特徴とする請求項2に記載の情報処理装置。

【請求項4】 CPUモジュールと、それぞれがMPUおよびRAMコアとを有する複数のメモリモジュールと、前記CPUとメモリモジュールとの接続、および／または、任意のメモリモジュール間の接続をなす複数組のバスとを備え、前記一以上のメモリモジュールにより把握される配列に関する処理が、CPUから各メモリモジュールのMPUに与えられるインストラクションにより、各メモリモジュールのMPUの作動により実行されるように構成された分散メモリ型情報処理システムを利用した配列のソート方法であって、(a)メモリモジュールにおいて、自己が把握する配列の部分構成する要素をソートするステップと、(b)前記自己の把握する前記部分が配列中に占める位置にしたがって、前記配列の部分把握するメモリモジュールのうち、要素および順位番号を送出する側の提示メモリモジュール、および、要素および順位番号を受理する側の判定メモリモジュールを決定するステップと、(c)提示メモリモジュールにおいて、ソートされた要素を、その順位番号とともに、他のメモリモジュールに所定のバスを介して伝達するステップと、(d)判定メモリモジュールにおいて、所定のバスを介して他のメモリモジュールからの前記要素および順位番号を受理するステップと、(e)前記判定メモリモジュールにおいて、当該判定メモリモジュールが把握する要素の順位番号に基づき、受理した要素の順位番号の候補を示す仮想順位番号を算出して、当該仮想順位番号を、前記提示メモリモジュールに返送するステップと、(f)前記提示メモリモジュールにおいて、前記仮想順位番号を受理した場合に、当該仮想順位番号にしたがって、要素の順位番号を更新するステップと、(g)前記ステップ(d)～(f)が終了することにより、当該ステップ(d)～(f)により所定の順位番号が付された要素に関する提示メモリモジュールと判定メモリモジュールからなるメモリモジュール群を、それぞれ、提示メモリモジュール群、および、判定メモリモジュール群の一方として、ステップ(d)～(f)を繰り返すことにより、各メモリモジュール群における要素の順位番号を更新することにより、配列の各要素の順位番

号を確定することを特徴とするソート方法。

【請求項5】 前記ステップ(e)が、(e1)受理した要素より前方に位置すべき要素の数を示す前方挿入数と、前方に位置すべき要素に関する順位番号と、受理した順位番号とに基づき、仮想順位番号を算出するステップを含むことを特徴とする請求項4に記載のソート方法。

【請求項6】 前記ステップ(f)が、(f1)受理した仮想順位番号を、ステップ(c)にて送出した要素の順位番号に代入するステップを含むことを特徴とする請求項4または5に記載のソート方法。

【請求項7】 さらに、(h)提示メモリモジュール群において、当該提示メモリモジュール群を構成するメモリモジュールにて把握されている要素が、当該メモリモジュール群においていくつ存在しているかを示す重複数を算出するステップを備え、

前記ステップ(c)が、(c1)同一の要素を重複して伝達しないように、ソートされた要素を、その順位番号および重複数とともに、他のメモリモジュールに伝達するステップを含み、前記ステップ(e)が、(e2)受理した要素より前方に位置すべき要素の数を示す前方挿入数と、前方に位置すべき要素に関する順位番号と、受理した順位番号および重複数に基づき、仮想順位番号を算出するステップを含み、かつ、前記ステップ(f)が、(f2)仮想順位番号と、ステップ(c)における要素の送出時の順位番号との差に基づき、当該要素と同一の要素の順位番号を決定するステップを含むことを特徴とする請求項4に記載のソート方法。

【請求項8】 初期的に提示メモリモジュールが単独のメモリモジュールであり、かつ、受信モジュールも単独のメモリモジュールであり、

(d)～(f)のステップが終了する毎に、 n ($n:1$ 以上の整数)がインクリメントされるような 2^n のメモリモジュールからなる提示メモリモジュール群と、 2^n のメモリモジュールからなる判定メモリモジュール群が形成されることを特徴とする請求項4ないし7の何れか一項に記載のソート方法。

【請求項9】 請求項4ないし6の何れか一項に記載された方法により、配列をソートし、かつ、当該ソートされた配列に基づき、前記配列中の要素が、重複なく、かつ、所定の順序にて配置された新たな配列を生成するコンパイル方法であって、(i)所定のメモリモジュールにおいて、順位番号にしたがって処理対象となる要素を送出するステップと、(j)前回の処理対象となった要素と同一の要素が送出された場合には、同一の要素の存在数を示す同一値個数カウンタをカウントアップし、その一方、前回の処理対象となった要素と異なる要素が送出された場合には、前回の処理対象となった要素、および、当該要素に関する同一値個数カウンタの値を関連付

けて、これらを送出するステップと、(k)前回の処理対象となった要素、および、関連する同一値カウンタの値を受理して、これらを関連付けて新たな配列中に配置するステップとを備え、(l)ステップ(i)～(j)を繰り返すことにより、前記新たな配列中に、要素およびその存在数が関連付けられて配置されることを特徴とするコンパイル方法。

【請求項10】 さらに、(m)何れかのモジュールにおいて、ステップ(j)にて送出される要素および関連する同一値個数カウンタの値をモニターするステップを備え、

当該何れかのモジュールにより、ステップ(k)が実行されることを特徴とする請求項9に記載のコンパイル方法。

【請求項11】 (n)当該配列の要素を把握するメモリモジュールにおいて、処理対象となっている要素の順位番号および当該要素の存在数をそれぞれ格納する順位番号カウンタおよび同一値個数カウンタを設けるとともに、および、前回の処理対象となった要素を一時的に格納するレジスタを設けるステップと、(o)順位番号にしたがって、当該順位番号が付された要素を把握するメモリモジュールにおいて、当該要素を第1のバスに送出するステップと、(p)配列の要素を把握するメモリモジュールにおいて、受理した要素とレジスタの内容とを比較して、これらが一致する場合には、存在数をカウントアップする一方、これらが一致しない場合には、レジスタの内容および存在数カウンタの値を、第2のバスに送出した後に、レジスタの内容および存在数カウンタの値を更新するステップと、(q)何れかのメモリモジュールにおいて、前記レジスタの内容および存在数カウンタの値を、それぞれ要素および当該要素の存在数として、配列中に配置するステップとを備えたことを特徴とする請求項9に記載のコンパイル方法。

【請求項12】 ステップ(n)が、さらに、(n1)処理対象となっている要素に関して、重複のない順位番号を格納する値カウンタを設けるステップを含み、前記ステップ(p)が、(p1)受理した要素とレジスタの内容とを比較して、これらが一致する場合に、当該処理対象となる要素の順位番号に、値番号カウンタの値を付与する一方、これらが一致しない場合に、値番号カウンタをカウントアップし、処理対象となる要素の順位番号に、カウントアップされた値番号カウンタの値を付与するステップを含むことを特徴とする請求項11に記載のコンパイル方法。

【請求項13】 請求項4ないし請求項8の何れか一項に記載のソート方法、および、請求項9ないし12の何れか一項に記載のコンパイル方法を用いて、複数の配列の共有化を実現する配列のジョイン方法であって、

(r)複数の配列を合併して、これら配列の要素の各々に順位番号を付す前記ソート方法にかかる処理を実行す

10

20

30

40

50

るステップと、(s) 前記合併した配列中の要素およびその順位番号にしたがって、前記コンパイル方法にかかる処理を実行し、重複した要素の存在しない新たな配列を生成するステップとを備えたことを特徴とする配列のジョイン方法。

【請求項 14】 CPU モジュールと、それぞれが MP U および RAM コアとを有する複数のメモリモジュールと、前記 CPU とメモリモジュールとの接続、および／または、任意のメモリモジュール間の接続をなす複数組のバスとを備え、前記一以上のメモリモジュールにより把握される配列に関する処理が、CPU から各メモリモジュールの MP U に与えられるインストラクションにより、各メモリモジュールの MP U の作動により実行されるように構成された分散メモリ型情報処理システムを利用した複数の配列のジョイン方法であって、請求項 4 ないし請求項 8 の何れか一項に記載のソート方法、および、請求項 9 ないし 12 の何れか一項に記載のコンパイル方法を用いて、複数の配列の共有化を実現する配列のジョイン方法において、

前記メモリモジュールが、それぞれ、レコード番号に基づき、要素を格納した配列である値リストにおける所定の要素を指定するために、レコード番号に対応する位置に、値リストを示すポインタ値を配置したポインタ配列を備え、(r1) 複数の値リストを合併して、これら配列の要素の各々に順位番号を付す前記ソート方法にかかる処理を実行するステップと、(t) 前記合併した値リスト中の要素およびその順位番号にしたがって、前記コンパイル方法にかかる処理を実行し、重複した要素の存在しない新たな値リストを生成するとともに、前記要素の順位番号を、重複した要素の存在しない場合の当該要素の順位番号に更新するステップと、(u) 前記重複した要素の存在しない場合の要素の順位番号からなる配列を、新たな値リストを示すための、新たなポインタ配列とするステップとを備えたことを特徴とするジョイン方法。

【発明の詳細な説明】

【0001】

【産業上の技術分野】本発明は、分散メモリ型の情報処理装置に関し、より詳細には、極めて高速にソート、コンパイルおよびジョインの処理を実現可能な情報処理装置に関する。

【0002】

【従来の技術】社会全体のさまざまな場所にコンピュータが導入され、インターネットをはじめとするネットワークが浸透した今日では、そこそこで、大規模なデータが蓄積されるようになった。このような大規模データを処理するには、膨大な計算が必要で、そのために並列処理を導入しようと試みるのは自然である。

【0003】さて、並列処理アーキテクチャは「共有メモリ型」と「分散メモリ型」に大別される。前者（「共

有メモリ型」）は、複数のプロセッサが 1 つの巨大なメモリ空間を共有する方式である。この方式では、プロセッサ群と共有メモリ間のトラフィックがボトルネックとなるので、百を越えるプロセッサを用いて現実的なシステムを構築することは容易ではない。したがって、例えば 10 億個の浮動小数点変数の平方根を計算する際、単一 CPU に対する加速比は、せいぜい 100 倍ということになる。経験的には、30 倍程度が上限である。

【0004】後者（「分散メモリ型」）は、各プロセッサがそれぞれローカルなメモリを持ち、これらを結合してシステムを構築する。この方式では、数百～数万ものプロセッサを組み込んだハードウェアシステムの設計が可能である。したがって、上記 10 億個の浮動小数点変数の平方根を計算する際の単一 CPU に対する加速比を、数百～数万倍とすることが可能である。

【0005】

【発明が解決しようとする課題】数百を越える多数のプロセッサによる並列処理の潜在的需要は大きいといわれているが、上述したように、現在の現実的なハードウェア技術でこれを実現しようとすると、分散メモリ型以外の手法による設計は困難である。分散メモリ型においては、個々のプロセッサに付属するメモリの容量が小さいため、並列処理の主たる目的の一つである大規模データ（通常は、配列）の保持および処理において、これを複数プロセッサおよびそれぞれに付属するメモリが分掌する必要がある。

【0006】しかしながら、複数プロセッサおよびそれぞれに付属するメモリが配列を分掌する場合には、バス上のデータの衝突を防止するためのバス調停が困難であり、各プロセッサが並列的に作動できなければ、プロセッサの利用効率を向上できず、その結果、処理の高速化を図ることができないなどの問題点がある。そこで、本発明は、以下のように、種々の目的を達成する。

【0007】(1) アルゴリズム的にバス上のデータの衝突が発生せず、バス調停が不要であり、これにより、バスのバンド幅をフルに生かして処理速度を向上させる。

(2) プロセッサ（好ましくは複数のプロセッサ）と目盛りとを備えたメモリモジュールを多数組み合わせ、これらによる並列処理を可能とし、それぞれのメモリモジュールを有効利用し、かつ、各メモリモジュール内のプロセッサに独立した処理を割り当てられるようにし、これにより、メモリモジュールの有効利用により処理速度をさらに向上させる。

(3) ソート対象のデータの大きさを「N」とした場合に、O(N) のデータの大きさしか必要としない。（従来のソート処理では、最悪の場合に O(N*N) や O(N*Log(N)) のデータ量を必要とし得る。）

(4) 処理時間が安定しており、最悪の場合でも、予想可能な処理速度が保証される。つまり、本発明は、著し

10

20

30

40

50

く高速に、かつ、安定した処理時間で、配列のソートなどが可能な情報処理装置を提供することを目的とする。

【0008】

【課題を解決するための手段】本発明の目的は、CPUモジュールと、それぞれがMPUおよびRAMコアとを有する複数のメモリモジュールと、前記CPUとメモリモジュールとの接続、および／または、任意のメモリモジュール間の接続をなす複数組のバスとを備え、前記一以上のメモリモジュールにより把握される配列に関する処理が、CPUから各メモリモジュールのMPUに与えられるインストラクションにより、各メモリモジュールのMPUの作動により実行されるように構成された分散メモリ型情報処理システムであって、前記メモリモジュールのMPUが、自己の把握する配列の部分を作成する要素のソートを実行して、前記要素を特定の順序にしたがって並べ替えるソート手段と、前記自己の把握する前記部分が配列中に占める位置にしたがって、前記ソートされた要素を、その順位番号とともに、他のメモリモジュールに所定のバスを介して伝達し、或いは、所定のバスを介して、他のメモリモジュールからの前記要素および順位番号を受理するI/Oと、前記要素および順位番号を受理した場合に、自己の把握する要素との比較により、受理した要素の順位番号の候補である仮想順位番号を算出して、前記他のメモリモジュールに返送する順位番号算出手段と、前記仮想順位番号を受理した場合に、当該仮想順位番号にしたがって、要素の順位を確定する順位確定手段とを備え、前記要素および順位番号を送出する側の提示メモリモジュールと、前記要素および順位番号を受理して仮想順位番号を算出する側の判定メモリモジュールとの通信により、前記配列の要素の順位番号を確定することを特徴とする情報処理システムにより達成される。

【0009】本発明によれば、提示メモリモジュールによる要素および順位番号の提示があるバスを介して実行され、判定メモリモジュールにより仮想順位番号が算出され、当該仮想順位番号が、他のバスを介して、提示メモリモジュールに与えられる。したがって、提示メモリモジュールおよび判定メモリモジュールにおいて、並列的にソート処理を進めることができ、かつ、バスの衝突も回避することが可能となる。

【0010】本発明の好ましい実施態様においては、前記メモリモジュールが、確定した順位番号にしたがって、処理対象となる要素を特定して何れかのバスに送出する要素特定／送出手段と、前回の処理対象となった要素と送出された要素とを比較する要素比較手段と、同一の要素が送出された場合には、その値をカウントアップする、同一の要素の存在数を示す同一値個数カウンタとを備え、前記要素比較手段が、前回の処理対象となった要素と送出された要素とが異なると判断した場合に、前回の処理対象となった要素、および、当該要素に関する

同一値個数カウンタの値を関連付けて何れかの送出するように構成され、さらに、何れかのメモリモジュールが、送出された前回の処理対象となった要素および関連するカウンタの値を受理して、これらを関連付け、かつ、受理した順序で配置する配列を備えている。この実施態様によれば、何れかのメモリモジュールにおいて、要素およびその重複数が、所定の順序で受理され、これにより、重複のない要素の配列、および、各要素の存在数を示す配列を作成することが可能となる。すなわち、これにより、重複のない要素のリスト、および、もとの配列において各要素の存在する数を容易に把握することができる。

【0011】本発明の別の実施態様によれば、前記メモリモジュールが、前記要素比較手段が、前回の処理対象となった要素と送出された要素とが異なると判断した場合に、その値をカウントアップする、重複のない順位番号を示す値番号カウンタと、送出された要素に関して、前回の処理対象となった要素と送出された要素とが同一の場合には値番号カウンタの値を、重複のない当該要素の順位番号と決定し、その一方、これらが異なる場合には、カウントアップされた値番号カウンタの値を、重複のない当該要素の順位番号と決定して、当該順位番号を更新する順位番号更新手段とを備えている。この実施態様によれば、配列の要素に付与された順位番号を、要素の重複を排除した状態のものに変換することが可能となる。

【0012】また、本発明の目的は、CPUモジュールと、それぞれがMPUおよびRAMコアとを有する複数のメモリモジュールと、前記CPUとメモリモジュールとの接続、および／または、任意のメモリモジュール間の接続をなす複数組のバスとを備え、前記一以上のメモリモジュールにより把握される配列に関する処理が、CPUから各メモリモジュールのMPUに与えられるインストラクションにより、各メモリモジュールのMPUの作動により実行されるように構成された分散メモリ型情報処理システムを利用した配列のソート方法であって、

(a)メモリモジュールにおいて、自己が把握する配列の部分を作成する要素をソートするステップと、(b)前記自己の把握する前記部分が配列中に占める位置にしたがって、前記配列の部分の把握するメモリモジュールのうち、要素および順位番号を送出する側の提示メモリモジュール、および、要素および順位番号を受理する側の判定メモリモジュールを決定するステップと、(c)提示メモリモジュールにおいて、ソートされた要素を、その順位番号とともに、他のメモリモジュールに所定のバスを介して伝達するステップと、(d)判定メモリモジュールにおいて、所定のバスを介して他のメモリモジュールからの前記要素および順位番号を受理するステップと、(e)前記判定メモリモジュールにおいて、当該判定メモリモジュールが把握する要素の順位番号に基づ

き、受理した要素の順位番号の候補を示す仮想順位番号を算出して、当該仮想順位番号を、前記提示メモリモジュールに返送するステップと、(f)前記提示メモリモジュールにおいて、前記仮想順位番号を受理した場合に、当該仮想順位番号にしたがって、要素の順位番号を更新するステップと、(g)前記ステップ(d)～

(f)が終了するごとに、当該ステップ(d)～(f)により所定の順位番号が付された要素に関する提示メモリモジュールと判定メモリモジュールからなるメモリモジュール群を、それぞれ、提示メモリモジュール群、および、判定メモリモジュール群の一方として、ステップ(d)～(f)を繰り返すことにより、各メモリモジュール群における要素の順位番号を更新することにより、配列の各要素の順位番号を確定することを特徴とするソート方法によっても達成される。

【0013】上記発明によれば、提示メモリモジュールにおける演算、提示メモリモジュールにおける要素および順位番号の送出、判定メモリモジュール群における演算、提示メモリモジュールにおける仮想順位番号の送出が、並列的に実行でき、かつ、バスの衝突も回避することができる。すなわち、これにより、著しく高速に、ソート処理(配列の要素への順位番号付与)を実現することが可能となる。また、使用するメモリ量も、 $O(N)$ に抑制することが可能となる。上記発明の好ましい実施態様においては、ステップ(e)が、(e1)受理した要素より前方に位置すべき要素の数を示す前方挿入数と、前方に位置すべき要素に関する順位番号と、受理した順位番号とに基づき、仮想順位番号を算出するステップを含む。また、さらに好ましい実施態様においては、ステップ(f)が、(f1)受理した仮想順位番号を、ステップ(c)にて送出した要素の順位番号に代入するステップを含む。

【0014】本発明の好ましい実施態様においては、さらに、(h)提示メモリモジュール群において、当該提示メモリモジュール群を構成するメモリモジュールにて把握されている要素が、当該メモリモジュール群においていくつ存在しているかを示す重複数を算出するステップを備え、前記ステップ(c)が、(c1)同一の要素を重複して伝達しないように、ソートされた要素を、その順位番号および重複数とともに、他のメモリモジュールに伝達するステップを含み、前記ステップ(e)が、

(e2)受理した要素より前方に位置すべき要素の数を示す前方挿入数と、前方に位置すべき要素に関する順位番号と、受理した順位番号および重複数とに基づき、仮想順位番号を算出するステップを含み、かつ、前記ステップ(f)が、(f2)仮想順位番号と、ステップ(c)における要素の送出時の順位番号との差に基づき、当該要素と同一の要素の順位番号を決定するステップを含む。

【0015】この実施態様によれば、提示メモリモジュールは、同じ要素を何度も送出する必要がない。また、

ある要素の重複数を算出すると、当該要素の順位番号および重複数を、判定メモリモジュールに送信し、判定メモリモジュールでは、当該要素にかかる仮想順位番号の算出を実行することができる。すなわち、これにより、メモリモジュールの利用効率を下げることを防止できる。

【0016】さらに好ましい実施態様においては、初期的に提示メモリモジュールが単独のメモリモジュールであり、かつ、受信モジュールも単独のメモリモジュールであり、(d)～(f)のステップが終了する毎に、 n ($n:1$ 以上の整数)がインクリメントされるような 2^n のメモリモジュールからなる提示メモリモジュール群と、 2^n のメモリモジュールからなる判定メモリモジュール群が形成される。上述したように、 2^n のメモリモジュールを用いると、好適に、ソート処理を実現することが可能となる。

【0017】また、本発明の別の実施態様においては、上記ソート方法により、配列をソートし、かつ、当該ソートされた配列に基づき、前記配列中の要素が、重複なく、かつ、所定の順序にて配置された新たな配列を生成するコンパイル方法は、(i)所定のメモリモジュールにおいて、順位番号にしたがって処理対象となる要素を送出するステップと、(j)前回の処理対象となった要素と同一の要素が送出された場合には、同一の要素の存在数を示す同一値個数カウンタをカウントアップし、その一方、前回の処理対象となった要素と異なる要素が送出された場合には、前回の処理対象となった要素、および、当該要素に関する同一値個数カウンタの値を関連付けて、これらを送出するステップと、(k)前回の処理対象となった要素、および、関連する同一値カウンタの値を受理して、これらに関連付けて新たな配列中に配置するステップとを備え、(l)ステップ(i)～(j)を繰り返すことにより、前記新たな配列中に、要素およびその存在数が関連付けられて配置されることを特徴とする。

【0018】また、上記コンパイル方法は、さらに、(m)何れかのモジュールにおいて、ステップ(j)にて送出される要素および関連する同一値個数カウンタの値をモニターするステップを備え、当該何れかのモジュールにより、ステップ(k)が実行されても良い。

【0019】また、上記コンパイル方法は、(n)当該配列の要素を把握するメモリモジュールにおいて、処理対象となっている要素の順位番号および当該要素の存在数をそれぞれ格納する順位番号カウンタおよび同一値個数カウンタを設けるとともに、および、前回の処理対象となった要素を一時的に格納するレジスタを設けるステップと、(o)順位番号にしたがって、当該順位番号が付された要素を把握するメモリモジュールにおいて、当該要素を第1のバスに送出するステップと、(p)配列の要素を把握するメモリモジュールにおいて、受理した

要素とレジスタの内容とを比較して、これらが一致する場合には、存在数をカウントアップする一方、これらが一致しない場合には、レジスタの内容および存在数カウンタの値を、第2のバスに送出した後に、レジスタの内容および存在数カウンタの値を更新するステップと、

(q) 何れかのメモリモジュールにおいて、前記レジスタの内容および存在数カウンタの値を、それぞれ要素および当該要素の存在数として、配列中に配置するステップとを備えても良い。

【0020】ステップ(n)は、さらに、(n1) 処理対象となっている要素に関して、重複のない順位番号を格納する値カウンタを設けるステップを含み、前記ステップ(p)は、(p1) 受理した要素とレジスタの内容とを比較して、これらが一致する場合に、当該処理対象となる要素の順位番号に、値番号カウンタの値を付与する一方、これらが一致しない場合に、値番号カウンタをカウントアップし、処理対象となる要素の順位番号に、カウントアップされた値番号カウンタの値を付与するステップを含むのが、さらに望ましい。

【0021】また、本発明の別の実施態様において、上記ソート方法および上記コンパイル方法を用いて、複数の配列の共有化を実現する配列のジョイン方法は、

(r) 複数の配列を合併して、これら配列の要素の各々に順位番号を付す前記ソート方法にかかる処理を実行するステップと、(s) 前記合併した配列中の要素およびその順位番号にしたがって、前記コンパイル方法にかかる処理を実行し、重複した要素の存在しない新たな配列を生成するステップとを備えている。すなわち、所望の配列を併合した状態で、本発明にかかるソート方法およびコンパイル方法を施すことにより、要素の重複を排除した、ジョインされた配列を得ることが可能となる。

【0022】さらに別の実施態様において、CPUモジュールと、それぞれがMPUおよびRAMコアとを有する複数のメモリモジュールと、前記CPUとメモリモジュールとの接続、および／または、任意のメモリモジュール間の接続をなす複数組のバスとを備え、前記一以上のメモリモジュールにより把握される配列に関する処理が、CPUから各メモリモジュールのMPUに与えられるインストラクションにより、各メモリモジュールのMPUの作動により実行されるように構成された分散メモリ型情報処理システムを利用した複数の配列のジョイン方法は、

【0023】上記ソート方法および上記コンパイル方法を用いており、かつ、メモリモジュールが、それぞれ、レコード番号に基づき、要素を格納した配列である値リストにおける所定の要素を指定するために、レコード番号に対応する位置に、値リストを示すポインタ値を配置したポインタ配列を備え、(r1) 複数の値リストを合併して、これら配列の要素の各々に順位番号を付す前記ソート方法にかかる処理を実行するステップと、(t) 前

記合併した値リスト中の要素およびその順位番号にしたがって、前記コンパイル方法にかかる処理を実行し、重複した要素の存在しない新たな値リストを生成するとともに、前記要素の順位番号を、重複した要素の存在しない場合の当該要素の順位番号に更新するステップと、

(u) 前記重複した要素の存在しない場合の要素の順位番号からなる配列を、新たな値リストを示すための、新たなポインタ配列とするステップとを備えている。

【0024】

【発明の実施の形態】[ハードウェア構成] 以下、添付図面を参照して、本発明の実施の形態につき説明を加える。図1は、本発明の実施の形態にかかるコンピュータシステムの構成を示すブロックダイアグラムである。図1に示すように、コンピュータシステム10は、単一命令による並列演算を実現するCPUモジュール12と、並列演算のために必要な種々のデータを記憶するメモリモジュール14-1、14-2、14-3、…と、必要なプログラムやデータを記憶する固定記憶装置16と、キーボードやマウスなどの入力装置18と、CRTなどからなる表示装置20と、種々の形式のデータ等が記憶されているレガシーメモリ22とを備えている。また、バス24-1、24-2、…において、CPUモジュール12、各メモリモジュール14との接点には、スイッチ28-1、28-2、28-3、…などが配設され、選択された回路要素間における情報の授受が可能となっている。また、CPUモジュール12とメモリモジュール14-1との間、隣接するメモリモジュール間において、バスの連結および接続をなすためのスイッチ30-1、30-2、…が設けられている。また、メモリモジュールの入力端子とバスとの接点と、当該メモリモジュールの出力端子とバスとの接点との間に、スイッチ(符号29参照)が設けられていても良い。図1において、上記スイッチは、破線の丸印にて示されている。

【0025】さらに、メモリモジュール14には、単一の入力端子および単一の出力端子だけではなく、一以上の他の端子(入出力端子など)が設けられているのが望ましい。たとえば、後述する第2の実施の形態や第3の実施の形態においては、3つ以上の端子からの入出力を用いて処理が実現されている。

【0026】CPUモジュール12と、メモリモジュール14との間には、複数のバス24-1、24-2、24-3、24-4、…とが設けられている。したがって、CPUモジュール12とメモリモジュール14との間、および、メモリモジュール間は、上記バスによりデータ等の授受が可能となっている。また、CPU12と、メモリモジュール14との間には、制御信号ライン25が設けられ、CPU12から発せられるインストラクションなどが、全てのメモリモジュール14に伝達されるようになっている。

【0027】さらに、CPU12と、他の構成要素(た

例えば、固定記憶装置16、入力装置18など）との間には、ローカルバス26が配設されており、これらの間でもデータ等の授受が可能となっている。CPU12は、固定記憶装置16に記憶され、或いは、バス26上に接続されたRAMのような他の記憶装置（図示せず）に記憶されたプログラムを読み出し、このプログラムにしたがって、以下に示すメモリモジュール14へのインストラクションの送出を含むデータの授受のほか、スイッチ28～30の制御等を実行する。また、CPU12は、プログラムにしたがって、レガシーメモリ22に記憶された種々の形式のデータを受け入れて、この形式のデータを、CPU12、メモリモジュール14、バス24からなる系にて処理可能な一連のデータ（配列）に変換し、これらを、各メモリモジュール14に記憶させることもできる。

【0028】図2は、各メモリモジュール14の概略を示すブロックダイヤグラムである。図2に示すように、メモリモジュール14は、CPUモジュール12から与えられるクロックなど同期信号を受け入れるクロックバッファ32と、データを記憶するRAMコア34と、後述する空間IDやデータの要素番号等を把握し、CPU12からのインストラクションなどを受理した場合に、空間IDや要素番号に基づき、RAMコア34へのデータ書き込みやRAMコアからのデータ読み出しを制御するMPU36と、バスの何れかからのデータを受け入れて、RAMコア34に供給し、および／または、RAMコア34からのデータを何れかのバスに送出するI/O38とを有している。この実施の形態において、メモリモジュール14は、制御信号ライン25を介して、CPUからのインストラクションを受け入れ、MPU36が、このインストラクションにตอบสนองして、RAMコア34のデータを読み出し、RAMコア34にデータを書き込み、或いは、データに所定の処理を施すことができるようになっている。また、RAMコア34へのデータアクセスや、I/Oを介してデータ入力およびデータ出力は、クロックバッファ32に与えられるクロックなどの同期信号に基づき実行される。上記メモリモジュール14のMPU36は、複数の処理ユニットからなり、並列的に複数の処理を実行できるのが望ましい。

【0029】図1および図2から明らかなように、本発明において、コンピュータシステム10は、メモリ共有型のシステムであると考えることができる。また、後述するように、制御信号ライン25を介して、各メモリモジュール14にインストラクションを与えることにより、各メモリモジュール14が並列的に処理を実行する。また、バスへのデータ出力およびバスからのデータ入力などが、所定の同期信号に基づき実行される。したがって、このコンピュータシステム10は、SIMDの形態をなしていると考えることができる。このような構成を備えたコンピュータシステム10は、基本的には、

本発明者の考案にかかる、特願平11-263793号に記載された多空間メモリ、メモリモジュールおよび組み替え可能バスを備えている。これらにつき、以下に簡単に説明を加える。

【0030】（1）多空間メモリ

本明細書において、多空間メモリとは、メモリ空間を、空間IDとアドレスとに基づきアクセスするために割り当てられたメモリ空間をいう。これにより、一連のデータが多数のプロセッサに分掌されていても、各プロセッサが、これを確実に分離、認識することができる。

【0031】従来のメモリ空間においては、プロセス毎に個別の領域を割り当てることはあっても、一連の変数（配列、構造体など）毎にメモリ空間を割り当てることは行われてこなかった。したがって、以下、このような従来のメモリ空間を「単一メモリ空間」と称する。単一メモリ空間のシステムにおいては、アドレスのみを用いてデータにアクセスしているため、関連を有する一連のデータを分離したり、認識することができなかった。このため、実際には並列処理が可能であっても、その可否を判断できない場合が多かった。また、ある単一メモリ空間に、新たな一連のデータを収容させる場合に、当該一連のデータの収容場所を確保するために、ガーベージコレクションを実行する必要があった。

【0032】これに対して、本発明においては、メモリ空間に、空間IDを導入し、一連のデータについて同一のIDを付与している。また、メモリモジュール14において、自身のRAMコア34に保持されているデータに関する空間IDを把握し、これにより、各メモリモジュール14自体が、現在アクセスされているデータの空間IDを参照することにより、自己の作動の是非を決定することができる。また、各メモリモジュールが空間IDと関連付けて、一連のデータの全部或いは一部を保持できるため、ある一連のデータを、複数のメモリモジュール14に分割して記憶させることができ、これによりガーベージコレクションを不要にすることができる。

【0033】（2）メモリモジュール

また、本発明においては、各メモリモジュール14が、MPU36を有し、上記空間IDのほか、自己が保持する一連のデータの各々の要素番号を把握している。したがって、CPU12からのインストラクションを受理した後、MPU36が、インストラクションにしたがってアクセスすべきデータが、自己のRAMコア34中に保持されているものか否かを判断して、アクセスに必要な是非を決定することができる。さらに、各メモリモジュール14が、自己のRAMコア34に格納されている配列要素の添え字の範囲から、SIMDでのインストラクションにおける暗黙の処理の分担範囲を決定することが可能である。各メモリモジュール14は、CPU12からのインストラクションにしたがって、処理すべき要素の記憶順序を入れ替えて、自己のRAMコア34中に保

持っている要素をソートすることが可能である。

【0034】(3) 組み替え可能バス

本発明においては、CPU12が、スイッチ28-1、28-2、…およびスイッチ30-1、30-2、…を選択的にオン/オフして、データの授受をなすべきメモリモジュール14を指定することにより、パイプライン処理を実現している。たとえば、図3に示すように、あるメモリモジュール14-iから出力されたデータを、他のメモリモジュール14-jに与え、かつ、当該他のメモリモジュール14-jから出力されたデータを、さらに他のメモリモジュール14-kに伝達すべき場合には、CPU12は、バス24-mを、メモリモジュール14-i、14-jのために割り当て、かつ、バス24-nを、メモリモジュール14-j、14-kのために割り当てるように、各スイッチの状態を設定する。

【0035】さらに、これらパイプライン処理は、単一のメモリモジュール間の接続により実現される場合だけでなく、複数の一連のメモリモジュール(メモリモジュール群)の間の接続により実現することも可能である。達成しようとする処理に応じて、各メモリモジュール間をつなぎ替え、各接続経路毎に、定められた種類のデータを定められた順序にて一方向に連続転送することで、バスの能力を100%近く使用できるように、通信をスケジューリング化することができる。これにより、分散メモリ型の並列処理システムの最大の問題であった、プロセッサ間通信のパフォーマンスの低さを、解消することができる。

【0036】[多空間メモリ] 再度、多空間メモリを用いた、本発明にかかるコンピュータシステムにおける各メモリモジュールのメモリ管理、および、インストラクションにしたがったメモリアクセスにつき、より詳細に説明を加える。図4は、多空間メモリの下での、メモリモジュール14の構造を説明するための図である。図4(a)に示すように、メモリモジュール14中のRAMコア34には、空間ID管理テーブルが設けられる。これにより、メモリモジュール14のMPU36は、自己が保持するデータの空間ID等必要な情報を把握することが可能となる。

【0037】図4(b)に示すように、空間ID管理テーブルには、自己が保持するデータ群ごとの、空間ID、CPUの管理の下での、データ群の論理開始アドレス、データ群が割り付けられた領域のサイズ、RAMコア34中の物理開始アドレス、当該空間IDを有する一連のデータの全サイズ、および、アクセス制限を示すアクセス制限フラグが格納されている。アクセス制限フラグは、この実施の形態においては、読み出しのみ可能(R)、書き込みのみ可能(R)、読み書き可能(RW)の3つの状態を示すことができるようになっている。メモリモジュール14のMPU36は、ある空間IDを有するデータ群が与えられた際に、RAMコア34

中に当該データ群を収容すべき、1以上の領域を見出して、当該領域にデータ群をそのまま、或いは、2以上に分割して収容する。この際に、与えられた空間ID、論理開始アドレス、全サイズ、アクセス制限フラグとともに、実際にデータを収容したRAMコア中の論理開始アドレスや、割り付け領域サイズも、空間ID管理テーブルに記憶される。図4(c)は、図4(b)による空間ID管理テーブルにしたがったRAMコア36中のデータを示す図である。

【0038】[メモリアクセスの概略] このように構成されたメモリモジュール14へのアクセスにつき以下に説明を加える。図5に示すように、まず、CPU12が、空間IDおよび論理アドレス、並びに、必要なインストラクション(たとえば、データの書き込みや読み出し)を、制御信号ライン25を介して、全てのメモリモジュール14に伝達する。各メモリモジュール14においては、これに応答して、MPU36に設けられた空間コンパレータ52が、空間IDと、自己の空間ID管理テーブル上に保持されている空間IDとを比較して、同一のものを、自己が保持しているかを判断し、また、アドレスコンパレータ54が、論理アドレスについて、同様の判断を行う。次いで、メモリモジュール14のMPU36が、自己のRAMコア34に、インストラクションによる処理対象となるデータが保持されていると判断した場合には、アドレスカリキュレータ56が、空間ID管理テーブルを参照して、RAMコア34中の物理アドレスを算出し、処理対象となるデータを特定する。このようにして、データが特定された後に、MPU36は、CPU12から与えられたインストラクションに応じた処理(たとえば、データの書き込みや読み出し)を実行し、必要な場合には、データをCPU12に伝達する(図5(c)参照)。

【0039】[ソート処理(第1の実施の形態)] このように構成されたコンピュータシステム10にかかるソート処理につき説明を加える。なお、以下の説明において、本発明にかかるメモリモジュールが、MPU(プロセッサ)を備えたメモリモジュールであることから、PMM(Processor Memory Module)と称する。

【0040】理解を容易にするために、図6に示すように、4つのPMMが、それぞれ、2つの要素(名字)を保持している場合を考える。図6(a)に示すように、あるPMM(第1のPMM14-1)には、要素の添え字(すなわちレコード番号)が「0」である「やまもと」という姓と、添え字が「1」である「あべ」という姓とが保持されている。第2のPMM14-2には、添え字が「2」である「いとう」という姓と、添え字が「3」である「すぎもと」という姓とが保持されている。以下、第3のPMM14-3、第4のPMM14-4にも、それぞれ、図6(a)に示すような添え字に対応した姓が保持されている。これら要素からなる配列に

は、同一の空間IDが付され、各PMMのMPU36は、その空間ID管理テーブルを利用して、自己のRAMコア34が管理する要素の添え字（レコード番号）や実際に格納している物理アドレス等を管理している。

【0041】たとえば、CPU12から、制御信号ライン25を介して、この空間IDを有する配列のソートをするインストラクションが、各PMM14-1~14-4に与えられたと考える。図7は、本実施の形態にかかるソート処理の処理手順を示すフローチャートである。図7に示すように、インストラクション（たとえば、「ある空間IDを有する配列中の要素をソートせよ」というインストラクション）がCPU12により発行されると（ステップ700）、このインストラクションにตอบสนองして、各PMMにおいて、各PMMのMPU36は、制御信号ライン25を介して与えられたインストラクションを受理して、その内容を解釈し（ステップ701）、インストラクション中の「空間ID」を調べ（ステップ702）、自己のRAMコア34が保持するデータの空間IDに関連しているか否かを判断する（ステップ703）。ステップ703にてノー（No）と判断された場合には、処理を終了し、その一方、イエス（Yes）と判断された場合には、MPU36は、空間ID管理テーブルを参照して、当該空間IDに関するデータ群が書き込み可能な状態になっているかなど、必要なチェックを行う（ステップ704）。チェックによって異常があると判断された場合（ステップ705でイエス（Yes））には、MPU36は、制御信号ライン25を介してエラーが生じたことをCPU12に通知する。その一方、異常がない場合には、MPU36は、以下に述べるソート処理本体を実行する（ステップ707以下）。

【0042】まず、処理に関連するPMM14-1~14-4の各々は、自己の保持する要素のソートを実行する（ステップ707）。このソートは、実際に、各PMM14中の要素の入れ替えを伴う。より具体的には、MPU36が、自己のRAMコア34中に保持された要素を、クイックソートなど既知のソート手法を用いてソートする。図6（b）は、図6（a）に示す各PMM中の配列中の要素がソートされた状態を示す図である。なお、図6（b）に示すように、上記要素のソートにともなって、各要素の添え字（レコード番号）の配置も変更されていることに留意すべきである。

【0043】次いで、各PMM14のMPU36は、自己が保持／管理している配列中の要素の数だけ、順位番号を配置するための領域（順位番号領域）を確保し、各順位番号の初期値を与える（ステップ708）。図6（c）は、各PMMに関して、順位番号の初期値が与えられた状態を示す図である。このように、初期的に、順位番号は、各モジュール内にてソートされた要素内で付与される。次いで、隣接するペア間のマージおよび順位番号付与が実行される（ステップ709）。ステップ7

09において、まずCPU12が、バス24上のスイッチ28、30を制御して、ソート処理に関連するPMMのうち、所定のペアの一方の入力と他方の出力、並びに、一方の出力と他方の入力とを接続する。上記ペアは隣接している2つのPMM、隣接していない場合にも近傍に位置する2つのPMMからなるのが望ましい。たとえば、図1において、ソート処理に関連するものが、PMM14-1~14-4である場合には、PMM14-1および14-2をペアとして、かつ、PMM14-3および14-4をペアとするのが望ましい。CPU12は、たとえば、図8に示すように、バス24-1にPMM14-1の出力およびPMM14-2を接続するように、かつ、バス24-2にPMM14-1の入力とPMM14-2の出力とを接続するように、スイッチ28を制御し、かつ、バス24-1にPMM14-3の出力およびPMM14-4を接続するように、かつ、バス24-2にPMM14-3の入力とPMM14-4の出力とを接続するように、スイッチ28を制御する。さらに、CPU12は、さらに、PMM14-2とPMM14-3との間に配置されるバス24-1、24-2上のスイッチ30-5、30-6をオフにする。図8において、黒丸で表わしているものが導通している状態を示し、白丸で表わしているものが導通しないPMMと接続されていない状態を示している。また、他のものは他のPMM（図示せず）の状態に従っている。なお、図8の例では、バス24-1、24-2を、スイッチ30-5、30-6をオフにすることにより分割して、バスをより有効に利用していることが理解できるであろう。

【0044】このようにして、図9に模式的に示すように、CPU12によってPMM間の接続が規定されると、PMMのペア間での順位番号付与の処理本体が実行される。図10ないし図12は、理解を容易にするために図6にて示した配列に関する順位番号付与を模式的に示す図であり、図13は、より一般的な、PMMのペア間の順位番号処理を示すフローチャートである。図10ないし図12においては、PMM14-1およびPMM14-2における処理過程のみを示したが、PMM14-3およびPMM14-4における処理も、並列的に実行されている。なお、ここで、処理において、最初にデータを他方のPMMに与えるものを前半のPMMと称し、受理するもの（他方のPMM）を後半のPMMと称する。前半のPMMは、要素や順位番号を提示するため提示PMMということができ、その一方、後半のPMMは、提示された順位番号を判定するため判定PMMとすることができる。ペアのうち何れのPMMが前半のPMMとなっても良い。この例では、便宜的に、PMM14-1が前半のPMMとなり、PMM14-2が後半のPMMとなっている。

【0045】まず、前半のPMMにおいては、処理位置を示すポインタ（以下、「PUTポインタ」と称する）

を初期位置（ソートされた配列の部分において先頭つまり「0」番目の位置）に配置する。その一方、後半のPMMにおいて、以下に説明するように、前半のPMMから受理する要素と、まず比較すべき位置などを示すポインタ（以下、「比較ポインタ」と称する）を初期位置（ソートされた配列の部分において先頭つまり「0」番目の位置）に配置する（図10（a）、および、図13のステップ1301、1311参照）。本実施の形態において、後半のPMMにて使用する比較ポインタは、（X、Y、Z）という構造体配列の形態をとっている。ここに、Xは、比較すべき先頭位置（つまり、未比較の要素の先頭位置、以後「未処理位置」と称する）を示し、Yは、前半のPMMから受信した要素の総数を示し（以下、場合によって「前方挿入数」と称する。）、Zは、前半のPMMと後半のPMMとをマージして得られた仮想的な配列における、前半のPMMから与えられた要素の順位番号の案（以下、場合によって「仮想順位番号」と称する。）を示す。

【0046】次いで、前半のPMMのMPUにより、最初のデータ転送が実行される。このデータ転送では、PUTポインタが示す位置の要素が、バスを介して、後半のPMMに伝達される（図10（b）およびステップ1303、1312参照）。なお、ステップ1302の分岐では、2つのPMM間の処理では常にイエス（Yes）と判断されるが、これについては後述する。最初のデータ転送では、要素「あべ」が後半のPMMに伝達される。後半のPMMにおいては、後半のPMMに格納されている配列の部分において、伝達された要素「あべ」を挿入すべき位置を捜し出す（ステップ1313）。これは実際に値を挿入するのではなく、挿入すべき位置を捜し出せば良い。本実施の形態において、各PMMの配列の部分に格納された要素は、実際にソートされた状態で配置されている。したがって、挿入位置の検索は、バイセクション法（二分法）など、高速検索手法を用いて実現することができる。挿入位置を捜し出すことにより、順位が確定していない要素であって、挿入位置より前方に位置する要素の範囲（以下、「範囲1」と称する）を特定することが可能となる。なお、本実施の形態において、同じ要素があった場合には、前半のPMMの順位が優先するという取り決めをしている。したがって、前半のPMMから伝達された要素「あべ」が、後半のPMMにも存在する場合には、前半PMMに格納されている方の順位が優先（つまり、より小さな順位番号）となる。

【0047】本例では、前半のPMMから伝達された要素「あべ」は、後半のPMMが把握する配列の部分中、要素「あべ」の前方に位置することがわかり、これにより、範囲「1」に属する要素が存在しないことがわかる（図10（c）参照、および、ステップ1314においてイエス（Yes））。そこで、後半のPMMのMPUは、

伝達された要素「あべ」の順位番号として「0（すなわち先頭）」を、他方のバスを介して、前半のPMMに返送する（ステップ1315）。次いで、後半のPMMのMPUは、前方挿入数をインクリメントして「1」にするとともに、仮想順位番号をインクリメントして「1」にする（ステップ1316）。これは、前方のPMMから伝達された要素が一つ増加したため、前方挿入数をインクリメントする必要があるため、かつ、次の要素の順位番号は、少なくとも、今回与えたもの（この場合では「0」）をインクリメントする必要があるからである。後半のPMMから要素の順位番号（挿入位置）が与えられると（ステップ1332）、前半のPMMのMPUは、与えられた順位番号を、該当する要素の順位番号として格納し（ステップ1334）、次いで、PUTポインタをインクリメントする（図11（a）、および、ステップ1335参照）。このようにして、前半のPMM中のある要素の順位が確定する。

【0048】次に、前半のPMMのMPUは、PUTポインタが示す位置の要素「やまもと」を、バスを介して、後半のPMMに伝達する（図11（b）およびステップ1303参照）。後半のPMMにおいては、先に、要素「あべ」が伝達されたときと同様に、伝達された要素「やまもと」を挿入すべき位置を捜し出す（ステップ1313）。要素「やまもと」は、後半のPMMが把握する配列の部分中、要素「はら」の後方に位置することがわかる（図11（c）、および、ステップ1314にてイエス（Yes））。これにより、後半のPMMが把握する配列の部分において、要素「はら」およびその前方に位置する要素の数、並びに、各要素の順位を確定することができる。より詳細には、後半のPMMのMPUは以下の手順にて、上記要素の順位を確定させる。

【0049】まず、範囲「1」に含まれる要素に関する順位番号に、それぞれ、前方挿入数「Y」が加えられる（ステップ1317）。これにより、範囲「1」に含まれる要素の順位が確定する。前述した例では、要素「あべ」の順位番号は「0 + 1 = 1」、要素「はら」の順位番号は「1 + 1 = 2」となる。次いで、範囲1に含まれる要素のうち、最後尾の要素の順位番号が、仮想番号に代入されるとともに（ステップ1318）、未処理位置を、範囲1における最後尾の要素の次の要素の位置に変更する（ステップ1319）。上記例では、要素「はら」の順位番号「2」が、比較ポインタ（構造体配列）のZに与えられ、かつ、未処理位置が、「0」から「2」に変更される。これにより、構造体配列は（2、1、2）となる。このような処理の後、構造体配列中の、前方挿入数「Y」および仮想順位番号「Z」がインクリメントされる（ステップ1320）。これにより、構造体配列は、（2、2、3）となる（図11（d）参照）。ステップ1320にて得られた仮想順位番号が、ステップ1312にて受信した要素（上記例では、「や

まもと」)の順位番号となり、後半のPMMのMPUは、当該順位番号(上記例では「3」)を前半のPMMに伝達する(ステップ1321)。このような処理の後に、さらに、仮想順位番号がインクリメントされる(ステップ1322)。これは、次の要素の順位番号は、少なくとも、今回与えた順位番号よりも1つ大きなものとなるからである。

【0050】前半のPMMは、受理した順位番号を、該当する要素の順位番号として格納し、次いで、PUTポインタをインクリメントする。このようにして、前半のPMMにおける要素の順位番号が確定する。前半のPMMにおいて、未処理の要素が既に存在しない(すなわち、すべての要素について順位番号が確定し、PUTポインタの位置には要素が配置されていない)には、前半のPMMのMPUは、終了を示す値を、後半のPMMに伝達する(ステップ1306参照)。ここに、終了を示す値は、配列の最後尾の要素を示す値よりも大きな値である。後半のPMMは、上記終了を示す値の受理にตอบสนองして、略同様の処理(図13のステップ1312~1322)の処理を実行する。上記例では、終了を示す値の受理にもかかわらず、範囲「1」に含まれる要素が存在しないため、ステップ1315、1316を介してステップ1323に達し、処理を終了する(図12(b)参照)。

【0051】前半のPMMにおいては、終了を示す値の送出(ステップ1316参照)により、および、全ての要素の順位番号の確定(ステップ1336でイエス(Yes))により処理が終了する。上記処理と同様の手順にて、PMM14-3とPMM14-4との間でも、マージ処理が実行され、これにより、図12(c)に示すように各要素の順位番号が確定する。

【0052】2つのPMMにおける各要素の順序番号が確定すると、CPU12は、スイッチを切り換えて、各々が2つのPMMからなる、2つのPMM群の間を接続する。図14(a)および図14(b)は、それぞれ、図8に示すPMMにおける、2つのPMM群の接続の一例を示す図である。図14(a)においては、PMM14-1、14-2が、第1のPMM群を構成し、CPU12は、PMM14-3、14-4が第2のPMM群を構成し、PMM14-1、14-2の出力と、PMM群14-3の入力とが接続され、PMM14-3の出力とPMM14-4の入力とが接続され、かつ、PMM14-4の出力とPMM14-1、14-2の入力とが接続されるように、スイッチ28、30を制御する(図7のステップ709参照)。或いは、図14(b)に示すように、PMM14-1、14-2の出力が、PMM14-3、14-4と接続されるようにスイッチが制御されても良い。

【0053】図15(a)、(b)は、それぞれ、図14(a)、(b)を模式的に表わした図である。後に明

らかになるが、図15(a)において、PMM14-4からPMM14-1、14-2に与えられるデータ(図中、符号①参照)は順位番号を示し、PMM14-1、14-2からPMM14-3に与えられるデータ(図中、符号②参照)は要素を示し、かつ、PMM14-3からPMM14-4に与えられるデータ(図中、符号③参照)は、要素およびPMM14-3が算出した仮想順位番号を示す。また、図15(b)においても、各PMM間において授受されるデータ①、②は、図15(a)のものと同じであり、その一方、PMM14-3からPMM14-4に伝達されるデータ(符号③参照)は、PMM14-3が算出した仮想順位番号を示す。

【0054】上記図12に示すように、2つのPMMのペアにおける配列の部分およびこれらに含まれる要素の順位番号から、2つのPMM群におけるマージ処理および配列の順位番号を決定する処理(図7のステップ709参照)につき、説明を加える。なお、以下の説明では、図14(b)、図15(b)に示すバスの接続態様にしたがって、各PMMにて実行する処理を説明する。

【0055】まず、各々がPMM14-1およびPMM14-2(以下、「前半のPMM群」と称する。)において、PUTポインタを初期位置に配置する(ステップ1301)。なお、以後の処理では、PUTポインタは、前半のPMM群を構成するPMMにおいて、自己の掌握する要素が送出されるのにしたがって移動する。その一方、後半のPMMの各々は、比較ポインタを、その構造体配列を初期化するとともに、初期位置に配置する(ステップ1302)。次いで、前半のPMM群を構成する各PMMにおいては、現在、前半のPMM群を構成する各PMMは、どの順位番号の要素が送出されたかを把握している。なお、フローチャートでは、PUTポインタとして、送信時に利用する送信ポインタと受信ポインタと双方を用いているが、基本的にこれら送信ポインタおよび受信ポインタの移動は、後半のPMM群における処理時間を挟むが、僅かな時間差のみをもって行われている。たとえば、後述するように、あるPMMにおいて送信ポインタをインクリメントとした(ステップ1304参照)場合には、当該PMMは、受信処理においても、受信ポインタをインクリメントする(ステップ1335参照)。

【0056】前半のPMM群を構成する各PMMは、処理の対象となる要素の順位番号に基づき、当該要素が、自己が掌握しているものか否かを判断する(ステップ1302)。このステップ1302にてイエス(Yes)と判断された場合には、PUTポインタの指し示す要素を、バス24を介して、PMM14-3、14-4に伝達する(図13のステップ1302、および、図16(a)参照)。上記例では、まず、順位番号「0」である要素「あべ」が、PMM14-1から、PMM14-3およびPMM14-4に伝達される。この処理により、PM

M14-1においては、PUTポインタの位置が移動する(ステップ1304)。

【0057】PMM14-3およびPMM14-4は、それぞれ、要素を受信し(ステップ1312)、その要素を挿入すべき位置を捜し出し(ステップ1313)、範囲「1」に属する要素が存在するか否かを判断する(ステップ1314)。上記要素「あべ」に関しては、ステップ1314においてノー(No)と判断される。これにより、PMM14-3において、要素「あべ」の仮想順位番号は「0」となるため、この値を、PMM14-4に伝達する。PMM14-4においても、要素「あべ」の仮想順位番号は「0」となる。そこで、PMM14-4のMPUは、「 $\max(0, 0) = 0$ 」を、要素「あべ」の順位番号として、前方のPMM群にバスを介して返送する(図16(b)およびステップ1315参照)。次いで、PMM14-3、14-4においては、構造体配列中の前方挿入数(Y)および仮想順位番号(Z)が、それぞれインクリメントする(ステップ1316)。上記例では、これにより、それぞれの構造体配列が、(0、1、1)、(0、1、1)となる。

【0058】後半のPMM群から順位番号が与えられる(ステップ1331)、前半のPMM群を構成する各PMMは、現在処理中の要素(たとえば、要素「あべ」)が、自己が掌握するものであるかを判断する(ステップ1333)。要素「あべ」の順位番号が伝達された場合には、PMM14-1が、上記ステップ1333でイエス(Yes)と判断し、その位置の要素に対応する順位番号を、後半のPMM群から与えられたものに書き換える(図16(a)およびステップ1334参照)。同様に、前半のPMM群は、次の順位番号を付与された要素を、後半のPMM群に伝達する。上記例では、PMM14-2から、要素「あべ」が伝達され(図17(a)参照)、後半のPMM群の各々の仮想順位番号のうち大きなもの「 $\max(1, 1) = 1$ 」が、当該要素「あべ」の順位番号として、前半のPMM群に伝達される(図17(b)参照)。また、後半のPMM群を構成するPMM14-3、14-4において、構造体配列は、それぞれ、(0、2、2)、(0、2、2)となる(図17(b)参照)。

【0059】さらに、前半のPMM群は、次の順位番号を付与された要素を、後半のPMMに伝達する。上記例では、PMM14-2から、要素「はら」が伝達される(図18(a)参照)。PMM14-3において、要素「はら」は、当該PMM14-3が掌握する要素「たなか」よりも後ろであると判断される(ステップ1313参照)。したがって、PMM14-3においては、範囲「1」には、要素「さとう」および要素「たなか」が属するため、要素「さとう」および要素「たなか」の順位番号に、それぞれ、前方挿入数「Y(=2)」が加えられる。これにより、要素「さとう」の順位番号は「0+

2=2」、要素「たなか」の順位番号は「2+2=4」と決定される(ステップ1317参照)。次いで、PMM14-3のMPUは、構造体配列(現在の値は(0、2、2))の仮想順位番号Zに、範囲「1」中の末尾の要素の順位番号「4」を与え(ステップ1318参照)、未処理位置を進める(すなわち、Xの値を「0」から「2」にする)(ステップ1319参照)。さらに、PMM14-3のMPUは、構造体配列(現在の値は(2、2、4))の前方挿入数「Y」および仮想順位番号「Z」をインクリメントする(ステップ1321参照)。これにより、構造体配列は、(2、3、5)となる。PMM14-3における仮想順位番号「Z(=5)」は、バスを介してPMM14-4に伝達される。その後、PMM14-3のMPUは、構造体配列の仮想順位番号「Z」をインクリメントする(ステップ1322参照)。上記例では、ステップ1322を施すことにより、構造体配列は、(2、3、6)となる。

【0060】その一方、PMM14-4において、要素「はら」は、当該PMM14-4が掌握する要素「すぎもと」と「よしだ」との間に位置すると判断される(ステップ1313参照)。したがって、PMM14-4において、範囲「1」には、要素「すぎもと」が属するため、要素「すぎもと」の順位番号に、前方挿入数「Y(=2)」が加えられ、これにより、要素「すぎもと」の順位番号は、「1+2=3」と決定される(ステップ1317参照)。次いで、PMM14-4のMPUは、構造体配列(現在の値は(0、2、2))の仮想順位番号「Z」に、範囲「1」中の末尾の要素の順位番号「3」を与え(ステップ1318参照)、未処理位置を進める(すなわち、「X」の値を「0」から「1」にする)(ステップ1319参照)。さらに、PMM14-4のMPUは、構造体配列(現在の値は(1、2、3))の前方挿入数「Y」および仮想順位番号「Z」をインクリメントする(ステップ1321参照)。これにより、構造体配列は(1、3、4)となる。

【0061】この後に、PMM14-4は、PMM14-3から与えられた仮想順位番号「Z(=5)」と、自己が算出した仮想順位番号「Z(=4)」とを比較し、より大きな方の値である「 $\max(5, 4) = 5$ 」を、伝達された要素「はら」の順位として、前半のPMM群に伝達する(ステップ321参照)。これにより、前半のPMM群において(より詳細には、要素「はら」を送出したPMM12-2において)、当該要素の順位番号が「5」であることが確定する。なお、PMM14-4においても、ステップ1321の後に、構造体配列中の仮想順位番号「Z」がインクリメントされる(ステップ1322参照)。上記例では、構造体配列は、(1、3、5)となる。

【0062】同様に、前半のPMM群より要素「やまと」が送出される(図19(a))が、この場合の処理

10

20

30

40

50

も、図13にしたがって実行される。再度、簡単に説明すると、要素「やまもと」を受理したPMM14-3においては、要素「やまもと」の挿入位置より前方に、範囲「1」に属する要素が存在しないため、PMM14-3は、その構造体配列中の仮想順位番号「Z(=6)」を、PMM-4に伝達する。PMM14-4においても、要素「やまもと」の挿入位置より前方に、範囲「1」に属する要素が存在しないため、その構造体配列中の仮想順位番号「Z(=5)」と、伝達された仮想順位番号「Z(=6)」とを比較して、その大きな方(MA 10 $X(6, 5) = 6$)を、要素「やまもと」の順位番号として、前半のPMM群に返送する(図19(b)およびステップ1315参照)。前半のPMM群においては、要素「やまもと」を送出したPMM14-1が、要素「やまもと」に対応する順位番号を、受理した順位番号(=6)に書き換える。なお、PMM14-3においては、ステップ1316を経ることにより、その構造体配列は(2, 4, 7)となり、その一方、PMM14-4においては、ステップ1316を経ることにより、その構造体配列は(1, 4, 6)となる。

【0063】このようにして、前半のPMM群において全ての要素の送出手続きが終了すると、前半のPMM群を構成する、何れかのPMMが、終了を示す値を後半のPMM群に伝達する(ステップ1306参照)。後半のPMM群を構成する各PMMは、これを受理して、それぞれ、ステップ1312ないしステップ1323の処理を実行する。上記例では、PMM13-4においては、順位の確定していない要素「よしだ」が存在する。このため、PMM13-4においては、ステップ1314においてイエス(Yes)と判断され、範囲「1」に属する要素「よしだ」の順位番号に、前方挿入数「Y」を加えて「3+4=7」、得られた数「7」を、要素「よしだ」の順位番号とする。このような処理を経た後、後半のPMM群を構成する各PMMにおいて、ステップ1323にてイエス(Yes)と判断され、後半のPMM群における処理も終了する。

【0064】上記例では、4つのPMMに配列中の要素が格納されていたが、それ以上のPMM中に配列中の要素が格納されている場合には、さらに、4つのPMMを一群のPMMとして、各々が4つのPMMからなるPM 40 M群のペアを作成し、これらペアの間に、略同様の処理を実行すれば良い。たとえば、図20に示すように、1024個のPMMにて、ある配列中の要素が格納されていると考える。この場合には、まず、PMM1およびPMM2、PMM3およびPMM4、PMM5およびPMM6、…PMM1023およびPMM1024を、それぞれ連結し(PMM間の実線参照)、これら2つのPMM間で、要素の順位番号を確定し、次いで、PMM1およびPMM2を前半のPMM群、PMM3およびPMM4を後半のPMM群とするPMM群のペア、PMM5お

よびPMM6を前半のPMM群、PMM7およびPMM8(図示せず)を後半のPMM群とするPMM群のペア、…PMM1021およびPMM1022(図示せず)を前半のPMM群、PMM1023およびPMM1024を後半のPMM群とするPMM群のペアを形成して、各ペア間を連結し(破線参照)、これらペアを構成する2つのPMM群の間に、要素の順位番号を確定する。以下、4つのPMMを前半のPMM群、および、これに引き続く4つのPMMを後半のPMM群とするPMM群のペア(一点鎖線参照)、8つのPMMを前半のPMM群、および、これに引き続く8つのPMMを後半のPMM群とするPMM群のペア(点線参照)というように、各々が2ⁿのPMM群からなるPMM群のペアを順次形成し、これらの間で、要素の順位番号を確定する。最終的に、512個のPMMを前半のPMM群、それに引き続く512個のPMMを後半のPMM群とするPMM群のペアの間に、要素の順位番号を確定することにより、1024個のPMM中の要素すべての順位番号を確定することが可能となる。

20 【0065】このように、各々が2ⁿ個のPMMからなるPMM群のペアを形成して、ペアを構成するPMM群の各PMMに格納された要素の順位番号を順次確定することにより(図7のステップ709、710参照)、最終的に、全ての要素の順位番号が確定すると(ステップ710でイエス(Yes)、必要な場合には、上記順位番号付けにしたがった配列を再形成する処理が実行される(ステップ711)。この処理は必須ではないが、順位番号にしたがって要素が配置されているような配列を生成することにより、後に実行される情報処理をより高速に実現することが可能となる。

30 【0066】より詳細には、まず、CPU12は、各PMMの入力および出力があるバスに接続されるように、スイッチ28、30を制御する。図21は、PMMが4つである場合に、これらの間の接続を模式的に示す図である。次いで、PMM14-1~14-4のMPUは、確定した順位番号にしたがって、要素および順位番号をバス上に放出する。各MPUは、バス上に放出される要素およびその順位番号をモニターし、もともと自己のRAMコアにて分掌していた要素の添え字(レコード番号)と同一の順位番号を有する要素を取り込み、RAMコアの所定の領域に格納する。たとえば、添え字(レコード番号)「0」および「1」の要素をもともと自己のRAMコアに記憶していたPMMにおいては(たとえば、図10のPMM14-1参照)、順位番号「0」および「1」を付された要素を取り込み、これらを記憶すれば良い。このようにすれば、各PMMにおいて、実際にソートされた配列を分掌することが可能となる。なお、このように、ソートされた配列を形成する際にも、PMMのMPUは、必要な空間ID管理テーブルを作成 50 する。

【0067】或いは、図22に示すように、ソートされた配列を分掌するための他のPMM（PMM14-5～PMM14-8）を設け、他のPMM群の各々が、PMM14-1～PMM14-4から、順次出力される要素およびその順位番号をモニターし、順位番号にしたがって、自己が取り込むべき要素を取り込んで、各PMMのRAMコアに記憶しても良い。たとえば、上記本発明を利用して、1024個のPMMを設け、各PMMに約100万のデータ（要素）を格納しておき、これらデータのソートを行った場合に、以下のような時間でソートが完了すると考えられる。ここに、各PMM間を接続するバスは、6.4GB/秒のデータ伝送が可能であり、かつ、処理中には、全てのPMMが並列的に動作し（すなわち、処理を実行していないPMMが存在せず）、かつ、関連するすべてのPMMが、同時にかつ協調的に同動作できると仮定する。また、各PMMにおける約100万のデータ（要素）のソートは、2.5秒にて完了すると考える。この場合、1024個のPMM中の、約10億個の要素をソートするために、略4秒程度しか必要としないことがわかる。

【0068】本実施の形態によれば、各PMMを、初期的には、2つのPMMのペアに分け、次いで、順次、各群が2ⁿ個のPMMから構成されるPMM群のペアに分けて、各ペアの間で順位番号を確定させていく。また、各ペアにて利用するバスを、スイッチ等を用いて調整することにより、各ペアにおける順位番号の確定が、並列的に実行することができる。さらに、前半のPMM群からの要素を、後半のPMM群に伝達し、後半のPMM群における構造体配列中の値にしたがって確定させ、確定された順位を、前半のPMM群に伝達する手順を繰り返すことにより、各ペアにおける順位番号の確定をなすことができる。したがって、処理を実行していない（いわゆる「遊んでいる」）PMMが生じることなく、極めて並列的に処理を実行できるとともに、バスを利用したデータ転送量を減じることができる。これにより、ソート速度を著しく高速にすることが可能となる。

【0069】なお、上記第1の実施の形態においては、図14（b）および図15（b）に示すようにPMMを接続し、これらの間で各要素に順位番号を付することにソート処理を実現しているが、図14（a）および図15（a）に示すようにPMMを接続しても良い。この場合には、図13における後半のPMM群の処理（ステップ1312～ステップ1323）が並列的に実行されず、あるPMMにおいて仮想順位番号が得られると、処理対象となる要素と当該仮想順位番号が、隣接するPMMに伝達され、当該PMMにおいて、ステップ1312～1323の処理が実行される。したがって、後半のPMM群を構成するPMMの個数が増えると、それだけ処理の遅延を招く場合もある。

【0070】〔他のソート処理（第2の実施の形態）〕

次に、本発明の第2の実施の形態につき説明を加える。上記第1の実施の形態においては、すべての要素（前半のPMM群内の要素）が、後半のPMM群に転送されている。しかしながら、配列が巨大になるにしたがって、重複値が多数出現し得る。上記第1の実施の形態にかかる手法では、同じ値をとる要素が何度もバス上に送出される。場合によっては、同じ値の要素を繰り返し送出することは、無駄であると考えることができる。そこで、第2の実施の形態においては、PMM群中の要素の個数を予めカウントし、要素とともにその個数を、後半のPMM群に送出することにより、重複する要素を繰り返しバス上に送出することを防止している。

【0071】たとえば、4つのPMMの対の各々においてソート処理が終了し、これら対を接続して、8つのPMMにおけるソート処理を実行することを考える。この場合に、図23に示すように、8つのPMMのマージおよびソート処理を実行するバス（図23においてPMMの下側に位置するバス、たとえば、符号2301～2303参照）のほか、他のバス（図23においてPMMの上側に位置するバス2304、2305参照）を用いて、PMM間のデータの授受ができるのが望ましい。図23に示すような接続態様において、PMM14-1～PMM14-4（以下、便宜上、「PMM14-1」ないし「PMM14-4」を、それぞれ、「PMM1」ないし「PMM4」と称する。）における値の重複数を算出する処理につき説明を加える。ここに、PMM1～PMM4の入出力端子（I/O）と接続されたバス（符号2304参照）を第1のバスと称し、PMM1～PMM4の他の入出力端子（I/O）と接続されたバス（符号2305参照）を第2のバスと称する。第1のバスは、PMM1～PMM4からなるPMM群の情報交換用に用いられ、第2のバスは、値およびその重複数を各PMMに与えるために利用される。

【0072】なお、以下の説明においては、図25に示すように、PMM1～PMM4中の配列において、各要素に順位番号が付されたものの重複数を算出している。すなわち、重複数は、前半のPMM群においてのみ算出すれば足りる。図24は、PMM群における重複数を算出するための処理を示すフローチャートである。PMM1～PMM4の各々は、まず、種々の初期化の処理を実行する（ステップ2401）。ここで、各PMMでは、処理にかかる値（要素）の順位番号を示す順位番号カウンタ、ある値（要素）がどれだけ重複して存在するかを示す同一値個数カウンタ、および、前回の処理において処理対象となった値（要素）を保持する前回値保存レジスタが設けられ、順位番号カウンタおよび同一値個数カウンタの値に初期値「0」が与えられる（図25参照）。なお、初期的には、前回値保存レジスタには何も値が保持されない。

【0073】次いで、各PMMは、順位番号カウンタを

10

20

30

40

50

参照して、処理対象となる要素の順位番号を特定し、当該順位番号を付された要素が、自己の掌握するものであるか否かを判断する(ステップ2403)。上記例では、初期的に順位番号カウンタのカウント値は「0」であるため、PMM3が、自己の掌握する要素が処理対象であると判断する(ステップ2403でイエス(Yes))。なお、次のステップ2404~2405は、最初の処理(すなわち順位番号「0」の要素に関する処理)では無視される。PMM3は、順位番号「0」が付された要素(この場合には「あべ」と同じ要素が、自分の中

にいくつ存在するか(すなわち、PMM3がいくつ「あべ」という要素を掌握しているか)を判定し、第1のパスに、要素「あべ」と、この要素をいくつ持っているかを示す自己PMM内存在数とを送出する(ステップ2406)。他のPMM(PMM1、PMM2およびPMM4)では、ステップ2403でノー(No)と判断されるため、ステップ2407に進む。

【0074】各PMMは、第1のパスを介して与えられたデータを受理し、データ中の自己PMM内存在数に基づき、順位番号カウンタのカウント値に、PMM内存在数を加える(ステップ2408)。上記例では、順位番号カウンタのカウント値が「0+1=1」となる。次いで、与えられた要素が、前回値保存レジスタのものと異なるか否かが判断され(ステップ2409)、双方が同一であった場合には、同一値個数カウンタのカウント値に、自己PMM内存在数が加えられ(ステップ2410)、その一方、新しい値のときには、後述する入替処理が実行される(ステップ2411)。なお、初回の処理では、前回値保存レジスタには何ら値が保持されていないため、上記ステップ2409の判断が省略され、かつ、前回値保存レジスタに、要素が収容されるとともに、同一値個数カウンタのカウントアップが実行される。したがって、上記例においては、各PMMは、受理した要素「あべ」を前回値保存レジスタに記憶するとともに、同一値個数カウンタを「0+1=1」とする(図26参照)。このようなステップ2401~2411の処理が繰り返され、最後の要素に関する処理が終了すると、ステップ2401においてイエス(Yes)と判断され、ステップ2412に進む。

【0075】上記例において、最初のステップ2401~2411の処理が終了すると、各PMMは、順位番号カウンタのカウント値を参照して、カウンタ値が「1」であることを確認する。これにより、PMM4が、順位番号「1」の要素を掌握していることがわかる。また、PMM4は、前回値保存レジスタの値(要素「あべ」と、順位番号「1」が付された要素「あべ」とを比較し(ステップ2404)、値に変化がないため、要素「あべ」および自PMM内存在数「1」を第1のパスに送出する(ステップ2405)。第1のパスを介してデータを受理した各PMMは、図27に示すように、順位番号

カウンタをカウントアップ(1+1=2)し(ステップ2408)、また、前回値保存レジスタに記憶された値と、受理した要素とが同一であるため、同一値個数カウンタをカウントアップ(1+1=2)する(ステップ2410)。

【0076】この後に、各PMMにおいては、順位番号「2」の要素に関する処理が実行される。順位番号「2」の要素の処理では、PMM1が要素を保持しているため、PMM1が、要素「いとう」と前回値保存レジスタに記憶された要素「あべ」とを比較する。ここでは、値に変化があるため(ステップ2404でイエス(Yes))、PMM1は、前回値保存レジスタの内容(要素「あべ」と、同一値個数カウンタの値「2」を第2のパスに送出する(ステップ2405)。このレジスタの内容およびカウンタ値は、各PMMに与えられる。後述するように、ある要素(この場合には要素「あべ」)の重複数が算出されると、当該要素に関するソート処理(図31参照)が実行され得る。したがって、各PMMにおいて、要素およびその重複数は、当該要素に関するソート処理が完了するまで保持していれば良い。また、要素「いとう」および自PMM内存在数「1」が第1のパスに与えられる(ステップ2406)。

【0077】各PMMは、第1のパスを介して与えられたデータに基づき、順位番号カウンタをカウントアップ(2+1=3)する(ステップ2408)。また、前回値保存レジスタの値「あべ」と、伝達された要素「いとう」とが異なるため(ステップ2409でイエス(Yes))、各PMMは、前回値保存レジスタの値を書き替える(更新する)とともに、同一値個数カウンタの値を、第1のパスを介して与えられた自PMM内存在数に置き換える(ステップ2411および図28(a)参照)。

【0078】他の順位番号の要素についても、同様の処理が施される。たとえば、順位番号「3」に関しては、PMM3が、ステップ2404、2406にしたがって、要素「いとう」を第1のパスに送出し、かつ、各PMMが、ステップ2407、2408、2409および2410にしたがって、各カウンタをカウントアップする(図28(b)参照)。また、順位番号「4」に関して、PMM1が、ステップ2404、2405、2406にしたがって、要素「いとう」および同一値個数カウンタのカウント値「2」を第2のパスに送出するとともに、要素「すぎもと」を第1のパスに送出し、かつ、各PMMが、ステップ2407、2408、2409、2411の順で、各カウンタをカウントアップするとともに、レジスタを更新する(図29(a)参照)。

【0079】順位番号「5」に関する処理では、PMM2は、自己が掌握する要素「すぎもと」が2つあることから、第1のパスに、要素「すぎもと」および自PMM内存在数「2」を送出する。したがって、各PMMにおいて、順位番号カウンタ、および、同一値個数カウンタ

のそれぞれのカウンタ値に、「2」が加えられる(図29(b)参照)。また、この処理により、順位番号カウンタのカウンタ値は、「5」から「7」に変化するため、次の処理対象となる要素の順位番号は「6」ではなく「7」になることに留意されたい。順位番号「7」を付された最後の要素に関する処理(図30(a)参照)が終了すると、ステップ2401にてイエス(Yes)と判断される。そこで、先頭のPMM(上記例では、PMM1)は、第2のパスに、要素「すぎもと」および同一個数カウンタのカウンタ値「4」を送出し(ステップ2413)、次いで、第2のパスに処理が終了したことを示すデータを送出する(ステップ2414)。各PMMには、各要素およびその個数を示す存在数が第2のパスを介して与えられ、これがソート処理に利用される。なお、上記例では、先頭のPMMが、ステップ2413、2414を実行するように構成したが、これに限定されるものではなく、予め、最後の要素等、および、終了を示すデータを出力するPMMを定めておけば良い。上述したように、あるPMM群における各要素の存在数を得ることにより、PMM群を、他のPMM群とマージして、これらの要素をソートする際に、重複した要素を送る必要がなくなる。

【0080】図31は、重複した要素の送出を排除したソート処理を示すフローチャートである。図31は、一部を除き、図13の処理と同一であり、その末尾2桁が同じものは、略対応する処理となる。また、図31において、二重の囲みを付した処理は、新規に追加された処理、或いは、図13の対応するものと若干異なる処理であることを示している。この処理においては、前半のPMM群において、処理対象となる要素(すなわち、送出ポイントにより指示される要素)を掌握するPMMは、その要素とともに、前半のPMM群における当該要素の重複数(存在数)「N」を、後半のPMM群に送出する(ステップ3103、3103-2参照)。たとえば、図25ないし図30に示した例において、PMM1~PMM4からなる前半のPMM群から、後半のPMM群に、要素「あべ」が送出される場合には、要素「あべ」のほか、前半のPMM群における要素「あべ」の重複数「2」が伝達される。また、前半のPMM群の送出処理において、要素およびその重複数を出力したPMMは、その出力の後に、自己が把握する当該要素の数だけ送出ポイントを移動させる(ステップ3104)。たとえば、図28(a)に示すように、要素「あべ」の重複数は「2」であり、これらがPMM3およびPMM4において、一つずつ把握されている。したがって、PMM3およびPMM4において、それぞれ、送出ポイントの位置が1つ下方に移動する。なお、各PMMにおける送出ポイントの移動量の総和が、当該要素の重複数「N」と等しくなる。

【0081】その一方、要素およびその重複数を受信し

た後半のPMM群を構成する各PMMにおいては、図31のステップ3116およびステップ3120に示すように、前方挿入数および仮想順位番号に、それぞれ、重複数「N」が加えられる。これは、自分より前方に位置する(順位が小さな)要素が、「N」だけ存在することに対応している。

【0082】さらに、前半のPMM群を構成するPMMの受信処理においては、前半のPMMによる送出処理において送出された要素(比較対象データ)と、後半のPMMによる処理において送出された順位番号とに基づき、受信した順位番号と、比較対象データの送出時における順位番号との差「M」が算出される(ステップ3132-2)。この差「M」は、後半のPMM群において、比較対象となっている要素の前方に位置する(すなわち、当該要素より小さい順位番号を付した)要素の数を示している。したがって、後半のPMM群を構成する各PMMは、自己の掌握する要素のうち、当該比較対象となっている要素と同一の要素を特定し(ステップ3132-3)、存在する場合には、これら要素の順位番号に、それぞれ「M」を加算する(ステップ3134)。ステップ3134の後に、PMMは、当該要素の数だけ受信ポイントを移動させる(ステップ3135)。この処理は、ステップ3104と略同様である。

【0083】次に、図24に示す重複数の算出と、図31に示すソート処理(場合により、「ソート本体」と称する。)との並列性につき説明を加える。図23に示すように、本実施の形態においては、重複数のカウントにかかるPMM間の通信を、バス2304、2305を利用し、ソート本体の実行にかかるPMM間の通信を、バス2301、2302、2303等を利用している。そこで、PMMにおいて並列処理が可能であれば、重複数のカウントとソート本体を並列して実行することができる。この場合に、前半のPMM群において、ある要素に関する重複数の算出が終了すると(たとえば、図28(a)に示すように、要素「あべ」およびその重複数「2」が第2のパスに送出され、前半のPMM群を構成するPMM(PMM1~PMM4)に受理されると、図31に示すような処理が、重複数の算出された要素に関して実行可能である。すなわち、ある要素の重複数の算出に応答して、当該要素に関するステップ3102~ステップ3104の処理、ステップ3112~ステップ3122の処理、および、ステップ3132~ステップ3135の処理を実行することができる。また、ある要素に関する要素とその重複数などは、上記図31に示す処理のうち、当該要素に関するものの終了とともに、削除することが可能である。したがって、前半のPMM群を構成するPMMの各々において、要素およびその重複数に関するデータ(その量は、異なる要素の数が多くなるに伴って大きくなる)を全て保持しておく必要もない。

【0084】このように、第2の実施の形態において

10

20

30

40

50

は、前半のPMM群において、重複数を算出し、要素およびその重複数を後半のPMM群に送出している。したがって、前半のPMM群が、後半のPMM群に同一の要素を重複して送る必要がなくなる。特に、同じ要素が数多く重複する場合（たとえば、要素が男女の種別を示すもの、年齢を示すものなど）には、ソート本体の処理回数を減少させることが可能となり、より高速にソート処理を実現することができる。

【0085】[コンパイル処理（第3の実施の形態）]
次に、本発明の第3の実施の形態につき説明を加える。第3の実施の形態では、各PMM内に配置された要素からなる配列に基づいて、レコード、各要素を重複なく配置した値リスト、および、レコードから値リストを指定するためのポインタ配列を作成する。この処理を、本明細書においてコンパイルと称する。たとえば、4つのPMM（PMM1～PMM4）に、ある配列の要素が分掌されている場合には、図32に示すように、PMMを接続すれば良い。図32に示すように、PMM1～PMM4の入出力端子（I/O）は、第1のバス（符号3201参照）により接続され、その一方、PMM1～PMM4の出力端子（O）および他のPMM“k”の入力端子（I）は、第2のバス（符号3202参照）により接続されている。

【0086】第1のバスは、PMM1～PMM4からなるPMM群の情報交換用に用いられ、第2のバスは、要素およびその重複数を他のPMM“k”に与えるために利用される。本実施の形態においては、上記要素およびその重複数に基づき、他のPMM“k”において、値リストおよび存在数配列等が形成される。なお、このPMM“k”は、PMM1～PMM4以外のPMMであっても良いが、無論PMM1～PMM4の何れかであっても良い。図33は、本実施の形態にかかるコンパイル処理を示すフローチャートである。なお、説明を容易にするために、図34（a）に示すように、PMM1～PMM4には、要素が分掌されており、これらの間で順位番号を付す処理が既に実行されていると考える。まず、各PMMにおいて、処理にかかる値（要素）の順位番号を示す順位番号カウンタ、処理の後の当該値（要素）の順位番号を示す値番号カウンタ、当該要素がどれだけ重複して存在するかを示す同一値個数カウンタ、および、前回の処理において処理対象となった値（要素）を保持する前回値保存レジスタが設けられ、各カウンタに初期値「0」が与えられる（ステップ3301および図34（a）参照）。なお、初期的には、前回値保存レジスタには値が保持されない。

【0087】以下、図33のステップ3302～ステップ3306の処理は、図24のステップ24021～2406と略同様である。すなわち、各PMMは、順位番号カウンタを参照して、処理対象となるようその順位番号を特定し、当該順位番号が付された要素が、自己の掌

握するものであるか否かを判断する（ステップ3303）。図34の状態では、順位番号カウンタのカウンタ値が「0」であるため、PMM3が、第2のバスに、当該PMM3が順位番号「0」を付された要素「あべ」をいくつ保持しているかを示す自己PMM内存在数（この場合には「1」）を創出する（ステップ3306および図34（b）参照）。次いで、PMM3は、前回値保存レジスタに記憶された要素と、第1のバスに放出した要素とが比較され、これらが相違する場合には、値番号カウンタのカウンタ値を、第1のバスに送出したようその順位番号に代入する（ステップ3307）。なお、図34の状態では、値番号カウンタのカウンタ値が初期値「0」であるため、要素「あべ」にかかる順位番号は変化しない（図34（b）参照）。

【0088】次いで、各PMMにおいては、第1のバスを介して与えられたデータを受理する（ステップ3308）。ステップ3308～3311の処理は、図24におけるステップ2408～2401の処理と略同様である。すなわち、各PMMは、順位番号カウンタのカウンタ値に、与えられたデータのPMM内存在数を加え、さらに、与えられたデータのうち、要素が新たなものではない場合（ステップ3310でノー(No)）には、同一値個数カウンタのカウンタ値に、PMM内存在数を加える（ステップ3311および図34（b）参照）。図34に示すように、順位番号「0」が付された要素「あべ」に関する処理が終了すると、順位番号「1」が付された要素に関する処理が、同様に実行される（図35（a）参照）。

【0089】さらに、順位番号「2」が付された要素に関する処理が実行される。ここでは、PMM1が、前回値保存レジスタに記憶された要素「あべ」と、順位番号「2」が付された要素「いとう」とを比較する。ここでは、これらが相違しているため（ステップ3304でイエス(Yes)）、PMM1は、第2のバスに、前回値保存レジスタに記憶された要素と、同一値個数カウンタのカウンタ値とを送出する（ステップ3305）。次いで、PMM1は、第1のバスに、処理対象の要素「いとう」と、PMM1が掌握する要素「いとう」の数である自己PMM内存在数「1」とを送出する（ステップ3306）。その後、PMM1は、前回値保存レジスタに記憶された要素と、第1のバスに放出した要素とを比較する。要素「いとう」を放出する場合には、これらが相違するため、要素「いとう」の順位番号に、値番号カウンタのカウンタ値に「1」を加えた値（ $0+1=1$ ）を代入する。各PMMは、第1のバスを介して与えられたデータを受理し（ステップ3308）、順位番号カウンタのカウンタ値に、受け入れたデータ中の自己PMM内存在数を加算する（ $2+1=3$ ）（ステップ3309および図35（b）参照）。要素「いとう」が与えられた場合には、前回値保存レジスタの要素「あべ」と与えられた

要素「いとう」とが異なるため（ステップ3310でイエス(Yes)）、各PMMは新値登録処理を実行する（ステップ3312）。この処理においては、値番号カウンタのカウント値がインクリメント（ $0+1=1$ ）され、同一値個数カウンタのカウント値が、受理したデータ中の自PMM内存在数「1」に変更され、かつ、前回値保存レジスタの内容が、要素「いとう」に書き換えられる（図35（b）参照）。

【0090】順位番号「3」の要素「いとう」に関しても同様の処理が施される。たとえば、PMM3は、第1のバスに要素「いとう」および自PMM内存在数「1」を送出し（ステップ3306参照）、かつ、当該要素「いとう」の順位番号に、値番号カウンタのカウント値「1」を代入する（ステップ3307および図36（a）参照）。また、各PMMは、順位番号カウンタのカウント値に、受理した自PMM内存在数「1」を加える（ステップ3309参照）とともに、同一値個数カウンタのカウント値に、自PMM内存在数「1」を加える（ステップ3311および図36（a）参照）。さらに、順位番号「4」の要素「すぎもと」に関しても、図36（b）に示すように、PMM1が、第2のバスに、要素「いとう」および同一値個数カウンタのカウント値「2」を送出し（ステップ3305）、第1のバスに、要素「すぎもと」および自PMM内存在数「1」を送出し（ステップ3306）、かつ、要素「すぎもと」の順位番号に、値番号カウンタに「1」を加えた値（ $1+1=2$ ）を代入する。その一方、各PMMにおいても、順位番号カウンタのカウントアップおよび新値登録処理が実行される（ステップ3309、3312および図36（b）参照）。他の順位番号の要素についても、同様の処理が施される。各要素についての処理は、図37（a）、（b）および図38に示されている。なお、図38に関して、PMM1は、最後の要素「すぎもと」、および、その要素の存在数を第1のバスに送出し、かつ、終了を示すデータを第2のバスに出力する（ステップ3315参照）。

【0091】前述したように、第2のバスには、PMM“k”の入力が接続されている。したがって、第2のバスには、重複のない要素と、これに関する値番号カウンタのカウント値とが与えられる。したがって、PMM“k”はこれらを受理し、受理した要素を、値リストに順次配置するとともに、受理した値番号カウンタのカウント値を、存在数配列に順次配置する。図39（a）は、PMM“k”内に作成された値リストおよび存在数配列を示す図である。これらは、ステップ3305或いはステップ3314にて送出され（図35（b）、図36（b）および図38参照）、PMM“k”に伝達されている。図39（a）に示すように、要素が重複なく値リストに配置され、かつ、各要素がいくつ存在するかを示す存在数（すなわち重複数）が存在数配列に配置され

ている。

【0092】さらに、PMM1～PMM4において、レコードと、各要素に、重複がないように付された順位番号とを対応させる値リストへのポインタ配列を作成することができる。つまり、レコードと、当該レコードに対応する要素に付された順位番号とを対応させた配列を作成すれば、これが値リストへのポインタ配列とすることができる（図39（b）参照）。図39（b）において、レコード「0」に関して、対応する要素の順位番号「2」が、値リストへのポインタ配列におけるポインタ値となる。これは、値リスト（図39（a）参照）において、格納位置番号「2」であるような値を指示すべきことを示している。すなわち、値リストへのポインタ配列のポインタ値により、PMM“k”に格納された値リストを指示することができ、これにより、レコードから要素を特定することが可能となる。

【0093】このように本実施の形態によれば、PMMに分掌された配列の要素をソートして順位番号を付し、かつ、同一の要素には同一の順位を付すように、順位を振り直している。要素は、新たに得られた重複のない順位と対応付けられて、値リストに格納される。当該順位は、値リストへのポインタ配列として、分掌された配列中の要素に対応つけられている。したがって、レコードに基づき、ポインタ配列のポインタ値を経て、値リスト中の要素を特定することが可能となる。

【0094】〔値リストの共有化（第4の実施の形態）〕次に、本発明の第4の実施の形態につき説明を加える。第4の実施の形態においては、二つの配列を共有化（ジョイン）している。この前提として、コンパイル処理による値リスト、および、値リストへのポインタ配列が作成されている。また、値リストや、値リストへのポインタ配列には、空間IDが付され、各PMMは、当該空間ID等により、自己が分掌している配列に関する種々の情報を把握している。

【0095】図40は、第4の実施の形態にかかる共有化処理を示すフローチャートである。説明を容易にするために、図41（a）に示すように、元のデータとして、レコードに対応した要素からなる配列（符号410参照）が、あるPMM群に分掌されていると考える。このレコード群に関するコンパイル処理により、PMM1およびPMM2からなるPMM群に、ポインタ配列（符号4101参照）および値リスト（符号4102参照）からなるブロック（以下、「情報ブロック」と称する。）が形成されている。その一方、元のデータとして、レコードに対応する要素からなる他の配列（符号4110参照）が、他のPMM群に分掌され、かつ、コンパイル処理により、PMM群3およびPMM群4からなるPMM群に、ポインタ配列（符号4111参照）および値リスト（符号4112参照）からなる情報ブロックが形成されていると考える。

【0096】各PMMに、CPU12から、値リストのジョインを指示するインストラクションが、二つの値リストを示す配列の空間IDとともに伝達される。各PMMのうち、ジョインすべき配列が自己の掌握する値リスト或いはその部分であるようなもの（つまり、上記例では、PMM1～PMM4）は、空間IDに基づき、ジョインの対象となる値リストを特定する（ステップ4001および図42（a）参照）。次いで、PMM1～PMM4は、二つの値リストを合併した状態で、これらをソートして各要素に順位番号を付与する（ステップ4002）。このソート処理のために、第1の実施の形態にかかるソート処理を利用することができる。上記例においては、まず、PMM1およびPMM2からなる第1のPMM群、および、PMM3およびPMM4からなる第2のPMM群のそれぞれにおいて、要素の順位番号付与の処理を実行し、次いで、第1のPMM群を前半のPMM群とし、かつ、第2のPMM群を後半のPMM群とすることで、二つのPMM群中の要素に順位番号を付与する。図42（b）は、このようにして要素に順位番号が付された状態を示す図である。

【0097】その後、処理対象となる値リストを分掌するPMMの間でコンパイル処理が実行され、これにより、他のPMM或いはPMM1～PMM4の何れかに、共通化された値リストおよび共通化された存在数配列が生成される（ステップ4003）。すなわち、コンパイル処理により、マージされた値リストの要素が重複しないような新たな値リストと、各要素がどれだけ重複して存在しているかを示す存在数を格納した存在数配列が得られる（図42（c）参照）。このような処理の後に、ジョインされた新たな値リスト（すなわちコンパイル処理により得られた値リスト）を指示するための新たなポインタ配列が求められる。これは、共有化前の情報ブロックにおけるポインタ配列中のポインタ値が示す、コンパイル処理により得られた順位番号配列の対応する順位番号を、当該ポインタ配列中のポインタ値の位置と対応する位置に格納するような新たなポインタ配列を作成することにより実現される。上記順位番号配列中の値は、各要素に付された新たな順位番号（図42（c）参照）に対応することは理解できるであろう。

【0098】図43（a）に示すように、たとえば、ポインタ配列中の第1のポインタ値「1」の示す位置の順位番号配列中の値（順位番号）は「2」であるため、共有化された後のポインタ配列中の対応する位置のポインタ値は「2」となる。また、第2のポインタ値「2」の示す位置の順位番号配列中の値（順位番号）は「3」であるため、共有化されたポインタ配列中の対応する位置のポインタ値は「3」となる。このようにして、ジョインされた値リストに関するポインタ配列を得ることが可能となる（図43（a）および図43（b））。

【0099】このような、新たなポインタ配列、およ

び、ジョインされた値リストにより、レコードから値（要素）を特定できることは明らかであろう。図44に示すように、レコードが、新たに得られた値リストへのポインタ配列中、対応する位置のポインタ値を特定し、かつ、当該ポインタ値が、その値が示す位置にある、値リスト中の要素を特定する。ここで、二つの値リストがジョインされているにもかかわらず、元のデータの要素と同一の要素が指定されることが理解できるであろう。

【0100】このように、第4の実施の形態によれば、複数の値リストを併合して、併合された値リストの要素に関して、ソート処理とコンパイル処理とを組み合わせることにより、ジョインされた値リスト、および、各値リストの順位番号配列を得る。レコードから値リストを指定するためのポインタ配列により順位番号配列の値（順位番号）が特定され、当該順位番号を、レコードに対応する位置に格納することにより、レコードに基づきジョインされた値リストを指定するための、新たなポインタ配列を得ることができる。したがって、上述したソート処理の時間およびコンパイル処理の時間程度で、複数の値リストをジョインすることが可能となり、著しく処理速度を向上させることが可能となる。

【0101】本発明は、以上の実施の形態に限定されることなく、特許請求の範囲に記載された発明の範囲内で、種々の変更が可能であり、それらも本発明の範囲内に包含されるものであることは言うまでもない。たとえば、前記実施の形態においては、本発明を、コンピュータシステムに適用しているがこれに限定されるものではなく、パーソナルコンピュータなどに接続可能なコンピュータボードに適用することもできる。この場合には、図1において、CPU12、メモリユニット14、バス24等がボード上に搭載され、これが、本発明における情報処理ユニットを構成する。

【0102】また、CPU12とメモリモジュール14との間、および／または、メモリモジュール14間を接続するバスの組の数は、前記実施の形態に限定されるものではなく、コンピュータシステムを搭載する回路基板の大きさ、各バスのビット数などを考慮して適宜決定することができる。また、前記実施の形態においては、メモリモジュールの入出力とバスとの接続を規定するためのスイッチ28と、CPUとメモリモジュールとの間、メモリモジュール間、或いは、メモリモジュールの入出力間で、バスの切断することができるスイッチ30とを設けている。スイッチ29、30を設けることにより、たとえば、あるバス（図1のバス24-4参照）を、CPUモジュール12とメモリモジュール14-1とのデータ授受のために利用するとともに、同時に、メモリモジュール14-2とメモリモジュール14-3との間のデータ授受のために利用することができる（この場合に、スイッチ30-5をオフにすれば良い）。したがって、より有効にバスを利用することが可能となってい

る。しかしながら、バスの組を数を十分に大きくできる場合、或いは、メモリモジュールの数が比較的少ない場合には、スイッチ29或いは30を必ずしも設けなくて良い。

【0103】また、本明細書において、制御信号ライン25を介して、CPU12からのインストラクションが与えられることを記載したが、制御信号ライン25を介して、インストラクションのほか、クロックなど、各メモリモジュールが同期して作動するための種々の制御信号が与えられ、かつ、各メモリモジュールからCPU12への所定の信号（たとえば、エラー信号や、データ受理を示す信号）が与えられていることは言うまでもない。

【0104】さらに、前記実施の形態において、PMM間の種々の接続を例示したが、PMM間の接続や送受信に利用するバスの選択は、上記実施の形態に示すものに限定されない。

【0105】また、前記第3の実施の形態においては、図32に示すように、第1のバス（符号3201）を利用して、各PMM間の通信をなし、かつ、第2のバス（符号3202）参照）を利用して、要素や当該要素の存在数（重複数）が通信されているがこれに限定されるものではなく、たとえば、図45に示すように、重複のない要素の配列である値リストやその存在数配列を生成するPMM“k”が、第1のバス4501をモニターして、第1のバス4501上に表れる要素や存在数配列に基づき、所定の処理（たとえば、PMM1～PMM4にて実行されるカウンタのカウントアップやレジスタの内容の保持／更新）を実行して、値リストや存在数配列を作成しても良い。

【0106】さらに、本明細書において、一つの手段の機能が、二つ以上の物理的手段により実現されても、若しくは、二つ以上の手段の機能が、一つの物理的手段により実現されてもよい。

【0107】

【発明の効果】本発明によれば、著しく高速に、かつ、安定した処理時間で、配列のソート、コンパイルおよびジョインが可能な情報処理装置を提供することが可能となる。

【図面の簡単な説明】

【図1】 図1は、本発明の実施の形態にかかるコンピュータシステムの構成を示すブロックダイアグラムである。

【図2】 図2は、本実施の形態にかかるメモリモジュールの概略を示すブロックダイアグラムである。

【図3】 図3は、本実施の形態にかかるメモリモジュール間のパイプライン処理を説明するための図である。

【図4】 図4は、本実施の形態にかかる多空間メモリの下での、メモリモジュール14の構造を説明するための図である。

【図5】 図5は、本実施の形態におけるメモリモジュールへのアクセスを説明するための図である。

【図6】 図6は、第1の実施の形態にかかるソート処理を施す配列の一例を示す図である。

【図7】 図7は、第1の実施の形態にかかるソート処理の処理手順を示すフローチャートである。

【図8】 図8は、第1の実施の形態にかかるソート処理を実施する際のメモリモジュール間の接続を示すブロックダイアグラムである。

10 【図9】 図9は、図8に示すメモリモジュール間の接続を模式的に示す図である。

【図10】 図10は、第1の実施の形態にかかるソート処理における配列中の要素への番号付与を示す図である。

【図11】 図11は、第1の実施の形態にかかるソート処理における配列中の要素への順位番号付与を示す図である。

20 【図12】 図12は、第1の実施の形態にかかるソート処理における配列中の要素への番号付与を示す図である。

【図13】 図13は、第1の実施の形態にかかるメモリモジュールのペア間の順位番号処理を示すフローチャートである。

【図14】 図14は、図8に示すメモリモジュールに関する、2つのメモリモジュール群の接続例を示すブロックダイアグラムである。

【図15】 図15は、図14に示す接続例を模式的に示す図である。

30 【図16】 図16は、第1の実施の形態にかかるソート処理における配列中の要素への順位番号付与を示す図である。

【図17】 図17は、第1の実施の形態にかかるソート処理における配列中の要素への順位番号付与を示す図である。

【図18】 図18は、第1の実施の形態にかかるソート処理における配列中の要素への順位番号付与を示す図である。

40 【図19】 図19は、第1の実施の形態にかかるソート処理における配列中の要素への順位番号付与を示す図である。

【図20】 図20は、第1の実施の形態にかかるソート処理におけるメモリモジュールの組み合わせを説明するための図である。

【図21】 図21は、第1の実施の形態にかかるソート処理の結果得られた順位番号にしたがって、新たな配列を生成する場合のメモリモジュールの接続例を示す図である。

50 【図22】 図22は、第1の実施の形態にかかるソート処理の結果得られた順位番号にしたがって、新たな配列を生成する場合のメモリモジュールの他の接続例を示す図である。

す図である。

【図23】 図23は、第2の実施の形態にかかるソート処理におけるメモリモジュールの接続例を模式的に示す図である。

【図24】 図24は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を説明するための図である。

【図25】 図25は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を説明するための図である。

【図26】 図26は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を示すフローチャートである。

【図27】 図27は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を説明するための図である。

【図28】 図28は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を説明するための図である。

【図29】 図29は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を説明するための図である。

【図30】 図30は、第2の実施の形態にかかるメモリモジュール群における重複数を算出するための処理を説明するための図である。

【図31】 図31は、重複した要素の送出を排除したソート処理を示すフローチャートである。

【図32】 図32は、本発明の第3の実施の形態にかかるコンパイル処理におけるメモリモジュールの接続例を模式的に示す図である。

【図33】 図33は、第3の実施の形態にかかるコンパイル処理を示すフローチャートである。

【図34】 図34は、第3の実施の形態にかかるメモリモジュール群におけるコンパイル処理を説明するための図である。

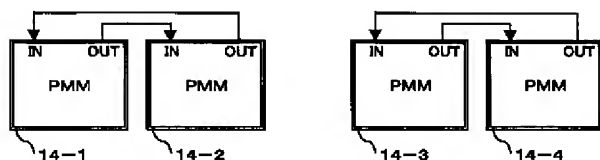
【図35】 図35は、第3の実施の形態にかかるメモリモジュール群におけるコンパイル処理を説明するための図である。

【図36】 図36は、第3の実施の形態にかかるメモリモジュール群におけるコンパイル処理を説明するための図である。

*

【図9】

図9



*【図37】 図37は、第3の実施の形態にかかるメモリモジュール群におけるコンパイル処理を説明するための図である。

【図38】 図38は、第3の実施の形態にかかるメモリモジュール群におけるコンパイル処理を説明するための図である。

【図39】 図39は、第3の実施の形態にかかるメモリモジュール群におけるコンパイル処理を説明するための図である。

10 【図40】 図40は、本発明の第4の実施の形態にかかる共有化処理を示すフローチャートである。

【図41】 図41は、第4の実施の形態にかかるメモリモジュール群におけるジョイン処理を説明するための図である。

【図42】 図42は、第4の実施の形態にかかるメモリモジュール群におけるジョイン処理を説明するための図である。

【図43】 図43は、第4の実施の形態にかかるメモリモジュール群におけるジョイン処理を説明するための図である。

20 【図44】 図44は、第4の実施の形態にかかるメモリモジュール群におけるジョイン処理を説明するための図である。

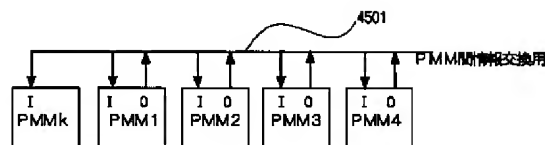
【図45】 図45は、本発明の他の応用例におけるメモリモジュール群の接続を概略的に示す図である。

【符号の説明】

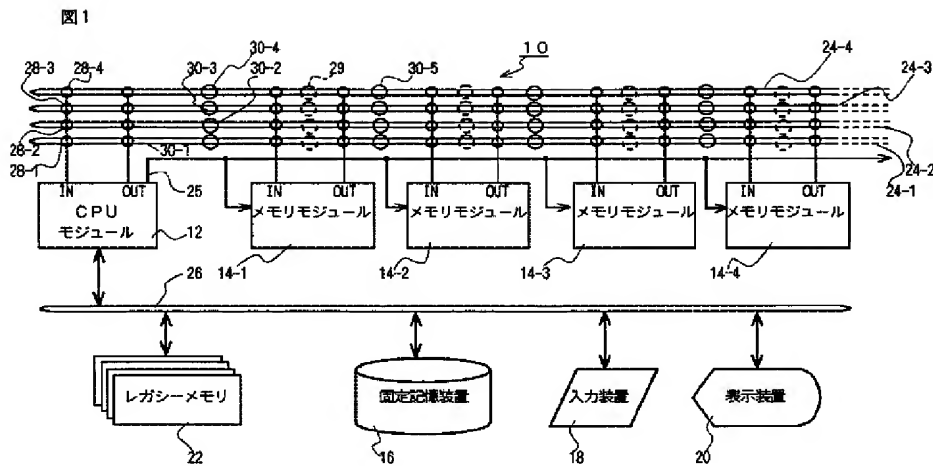
10	コンピュータシステム
12	CPUモジュール
14	メモリモジュール
16	固定記憶装置
18	入力装置
20	表示装置
22	レガシーメモリ
24、26	バス
25	制御信号ライン
28、29、30	スイッチ
32	クロックバッファ
34	RAMコア
36	MPU
38	I/O

【図45】

図45

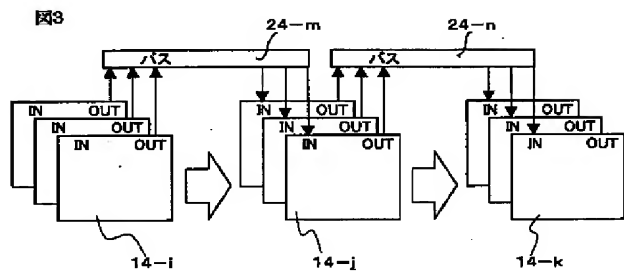
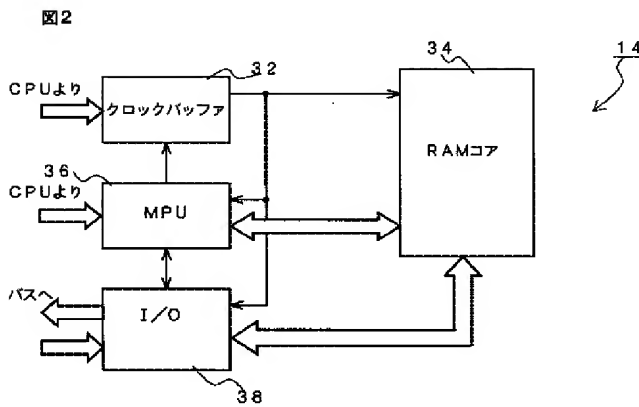


【図1】

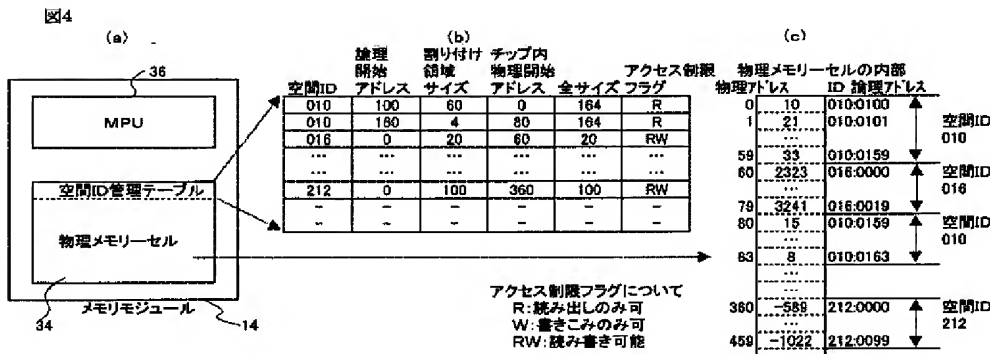


【図2】

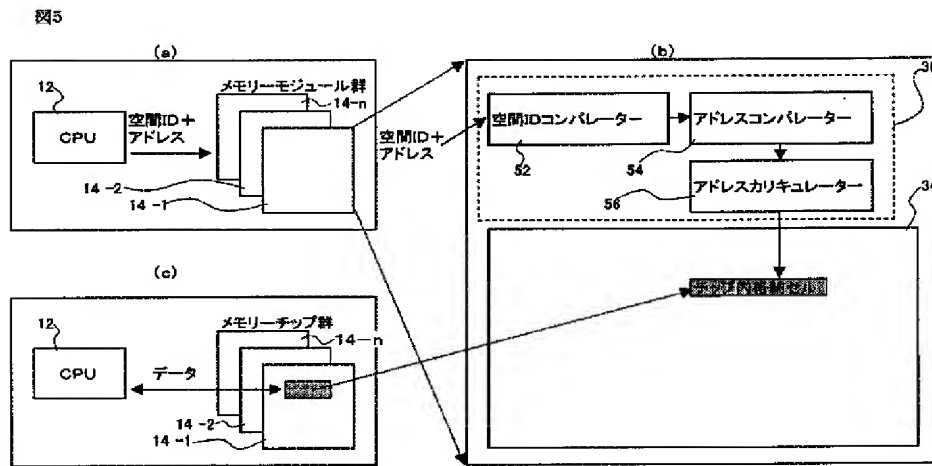
【図3】



【図4】

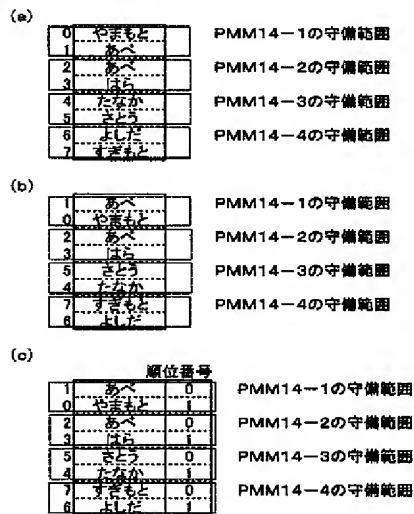


【図5】



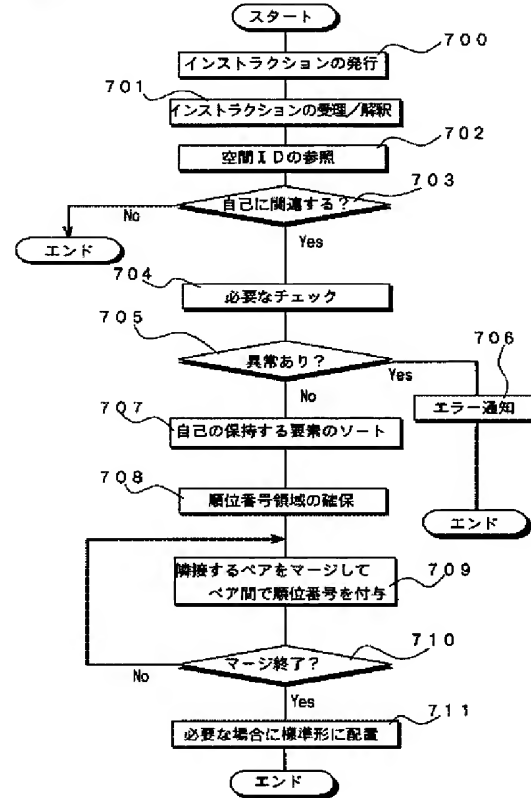
【図6】

図6



【図7】

図7



【図8】

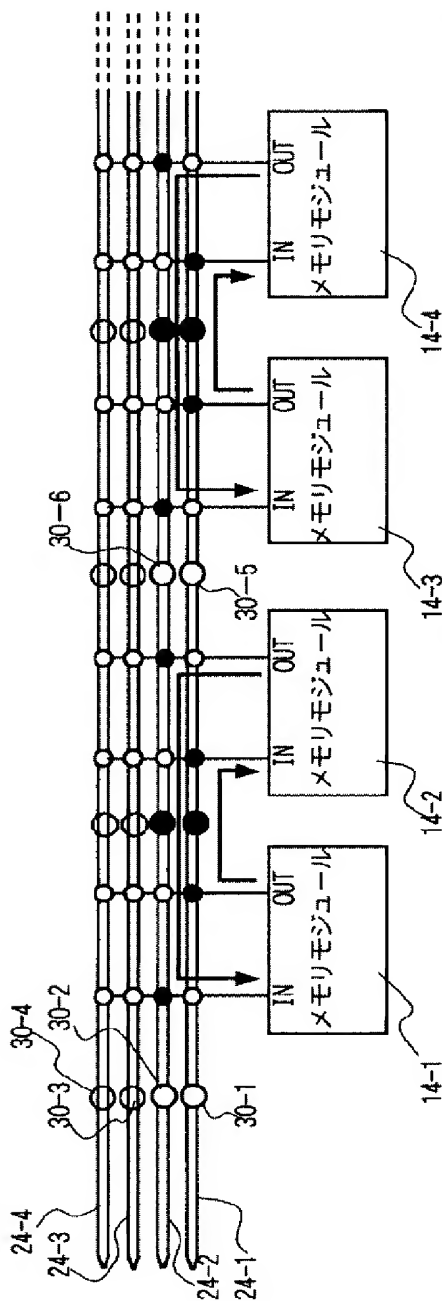
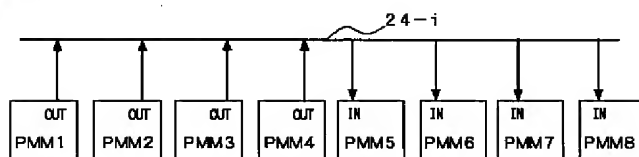


図8

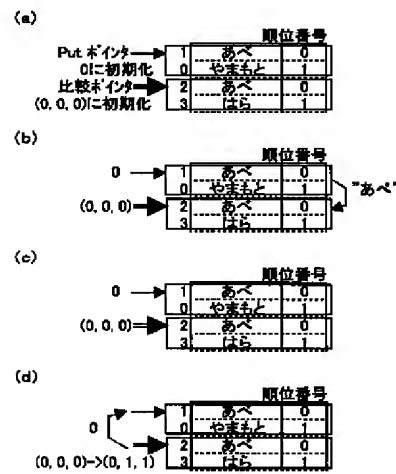
【図22】

図22



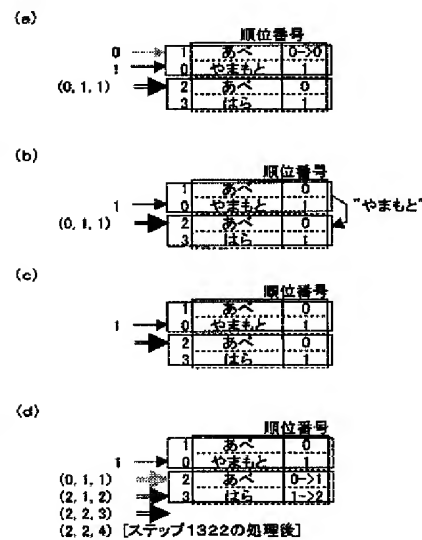
【図10】

図10



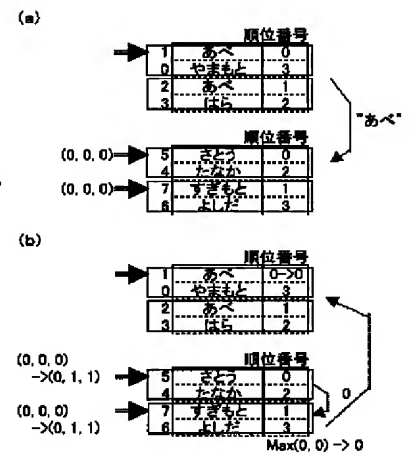
【図11】

図11



【図16】

図16



【図25】

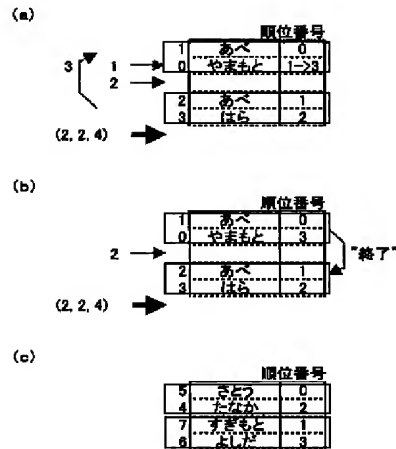
図25

準備完了状態
PMM1～PMM4までソート完了

順位番号	カウンタ	同一値回数	前回値保存レジスタ
1	0	0	-
2	0	0	-
3	0	0	-
4	0	0	-
5	0	0	-
6	0	0	-
7	0	0	-
8	0	0	-

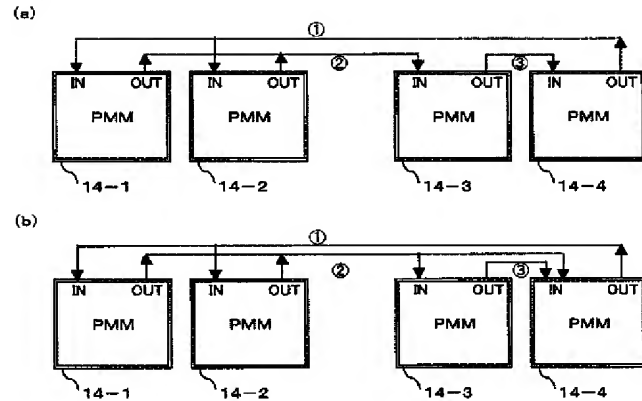
【図12】

図12

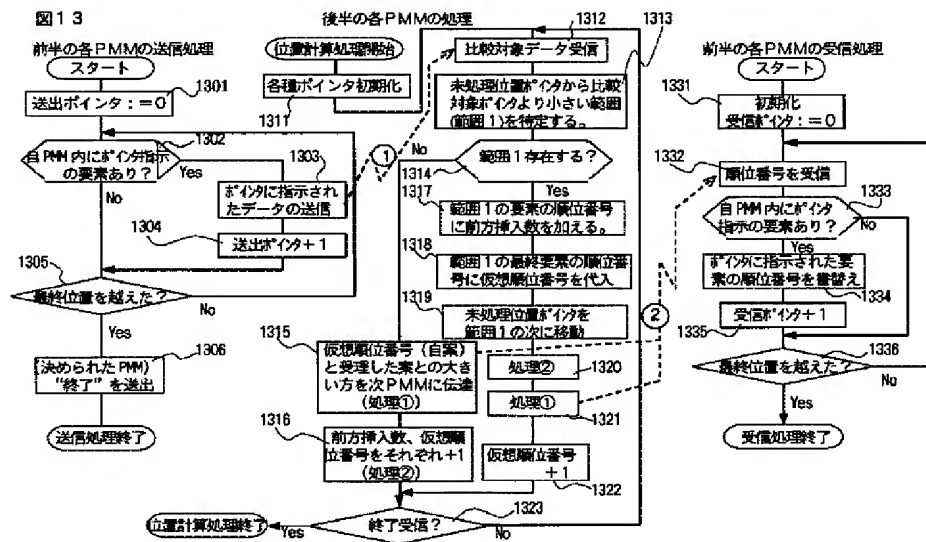


【図15】

図15

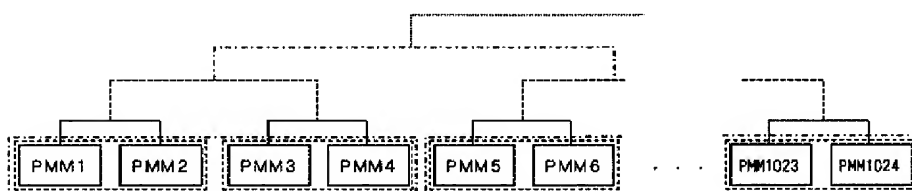


【図13】

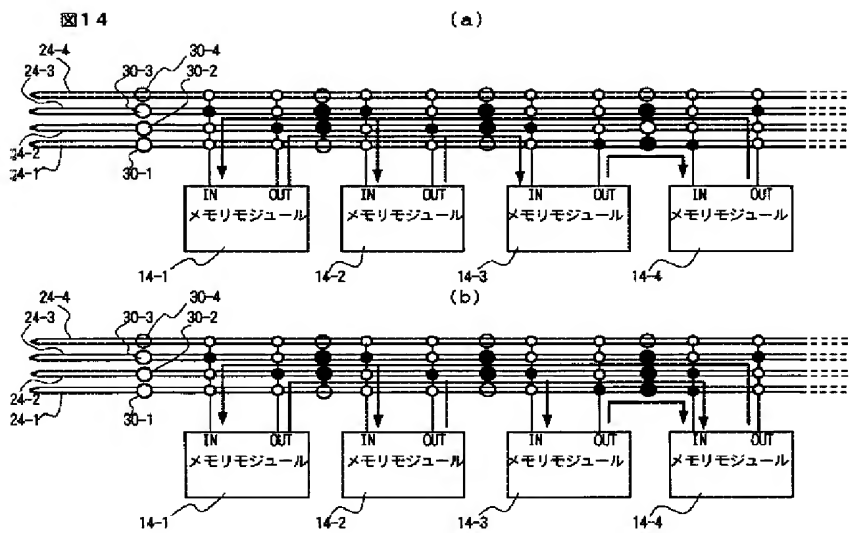


【図20】

図20

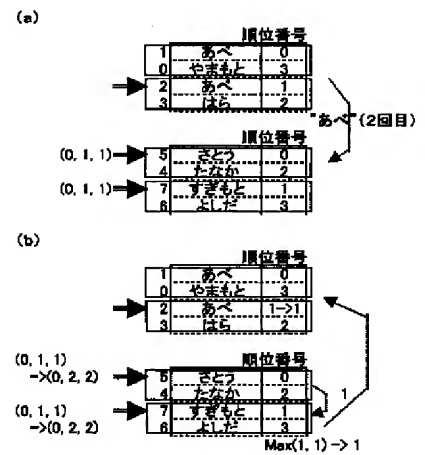


【図14】



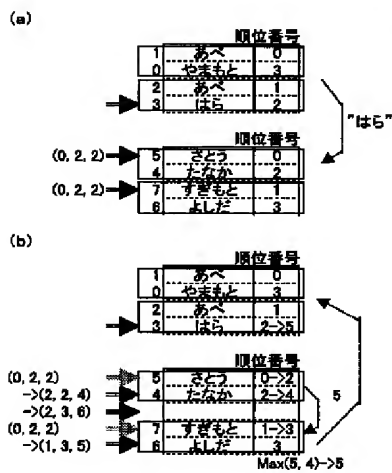
【図17】

図17



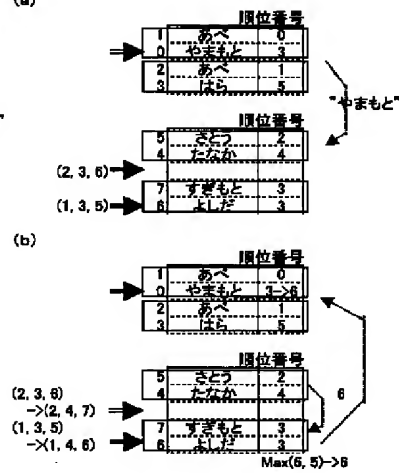
【図18】

図18



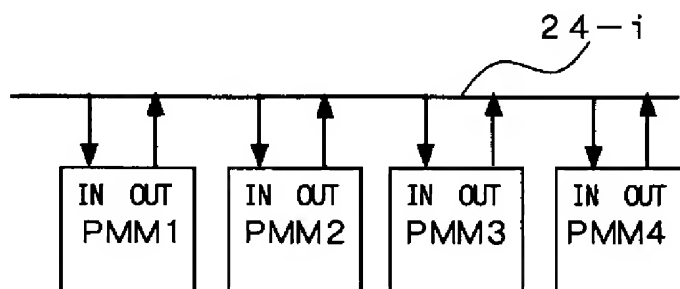
【図19】

図19



【図21】

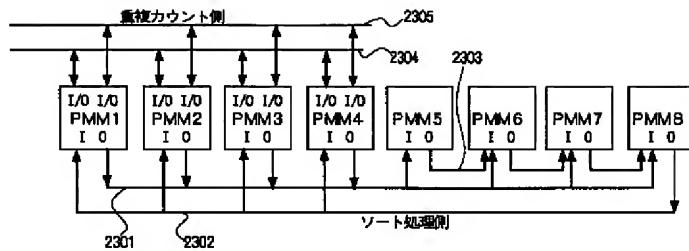
図21



【図23】

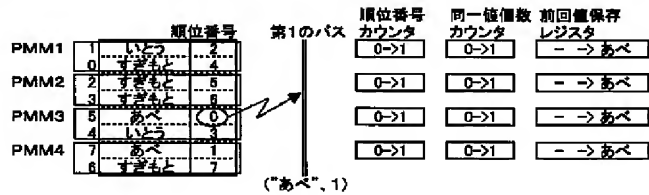
【図24】

図23



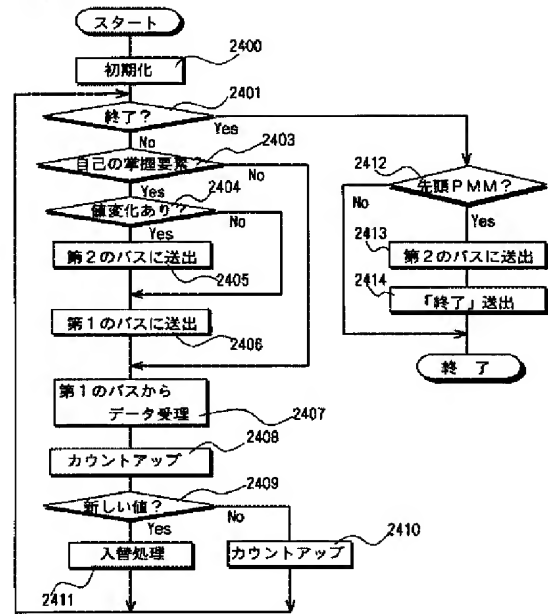
【図26】

図26



【図32】

図24



【図27】

図27

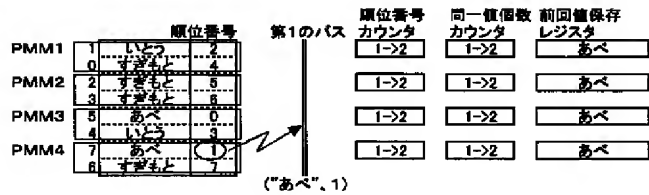
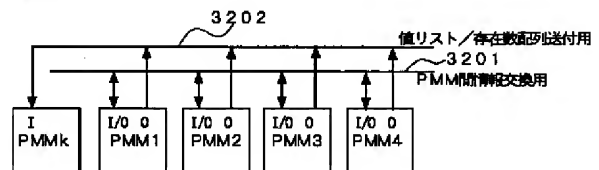
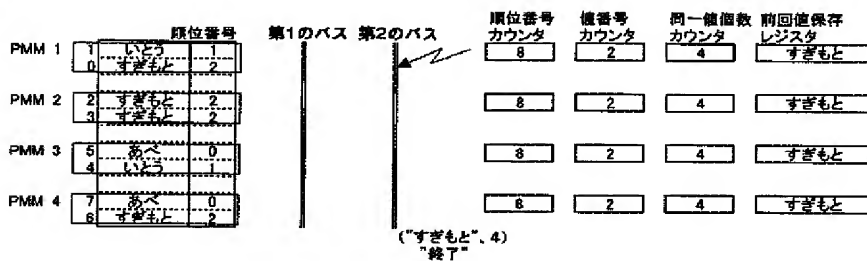


図32



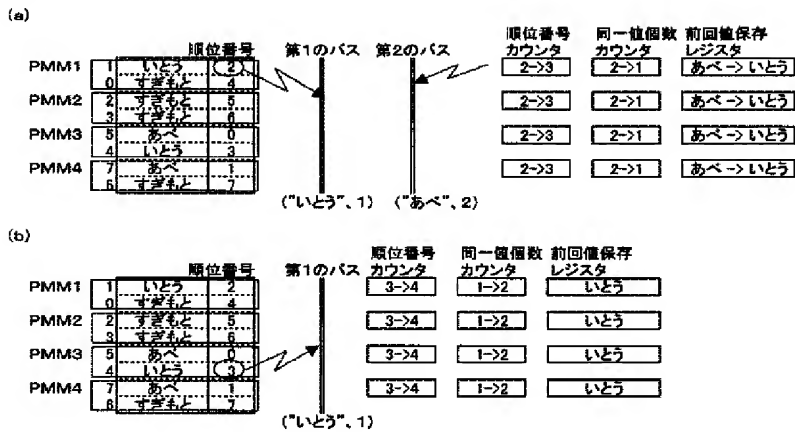
【図38】

図38



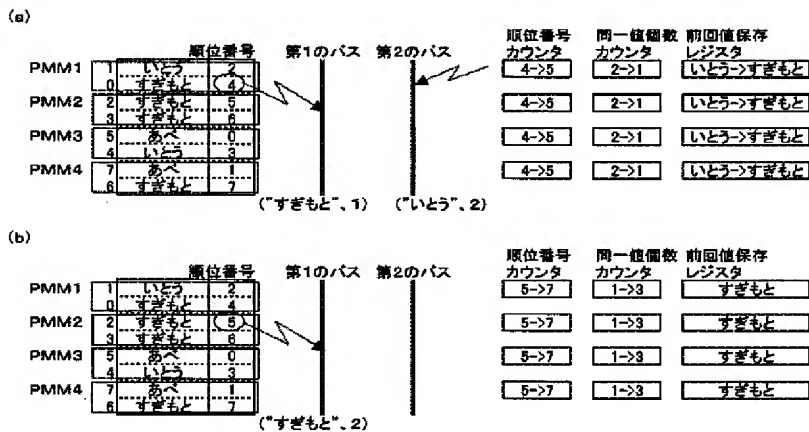
【図28】

図28



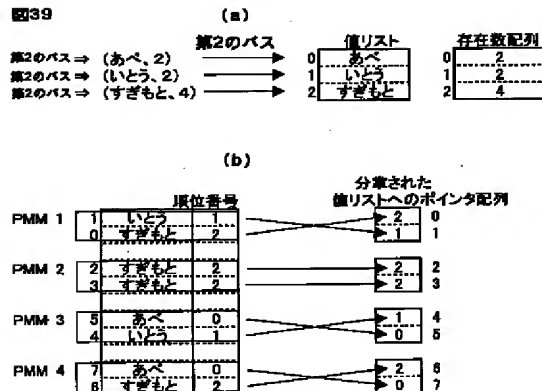
【図29】

図29



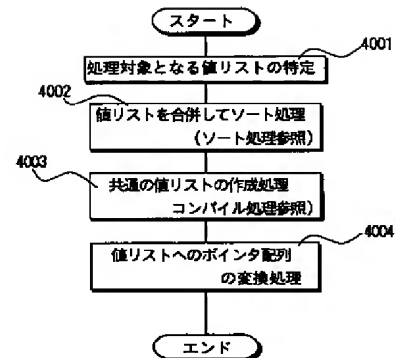
【図39】

図39



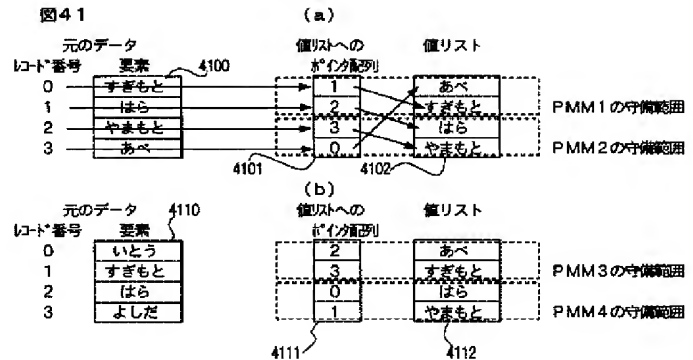
【図40】

図40



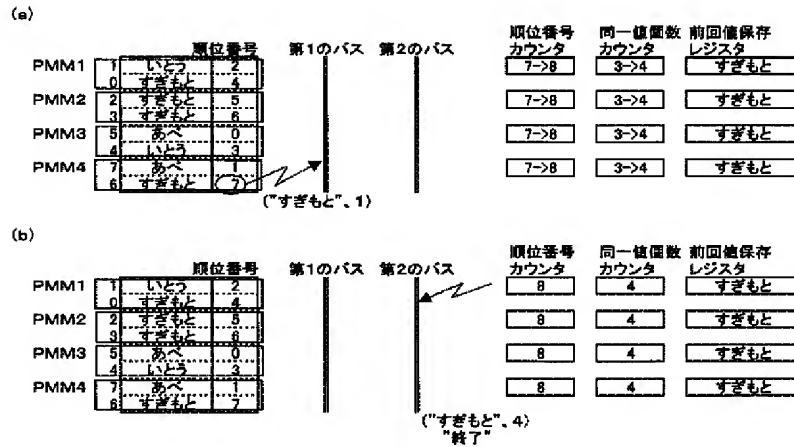
【図41】

図41

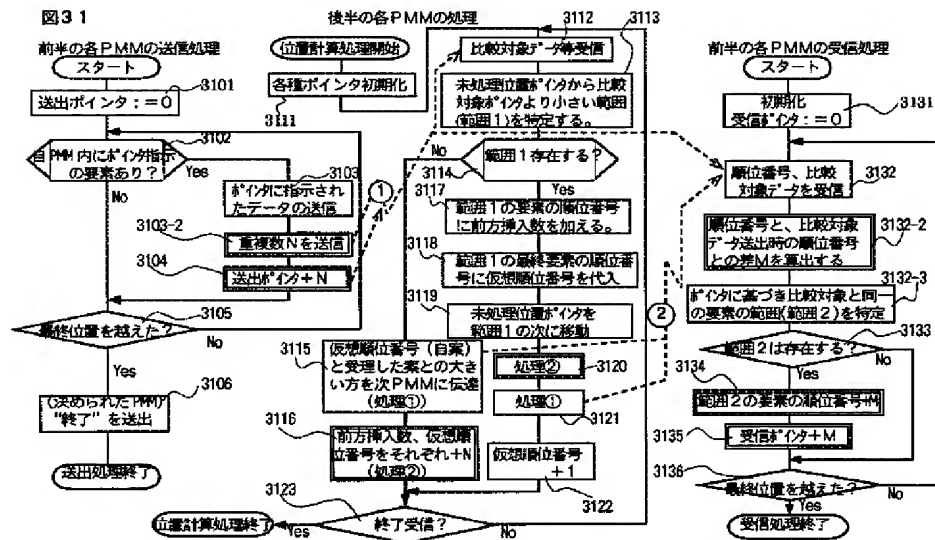


【図30】

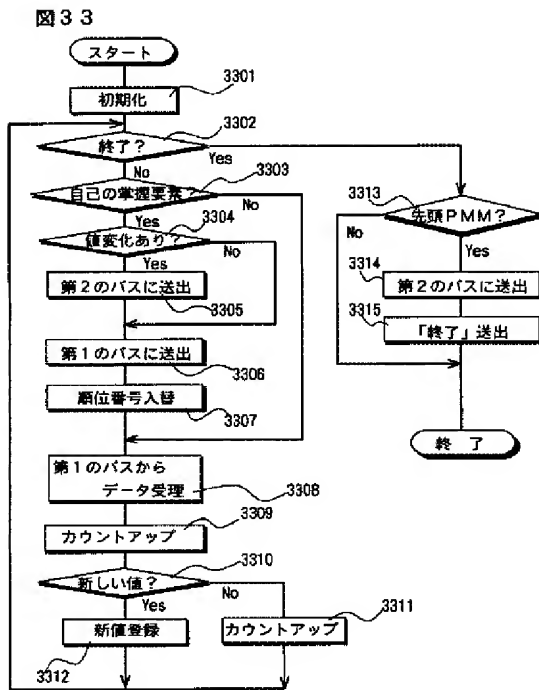
図30



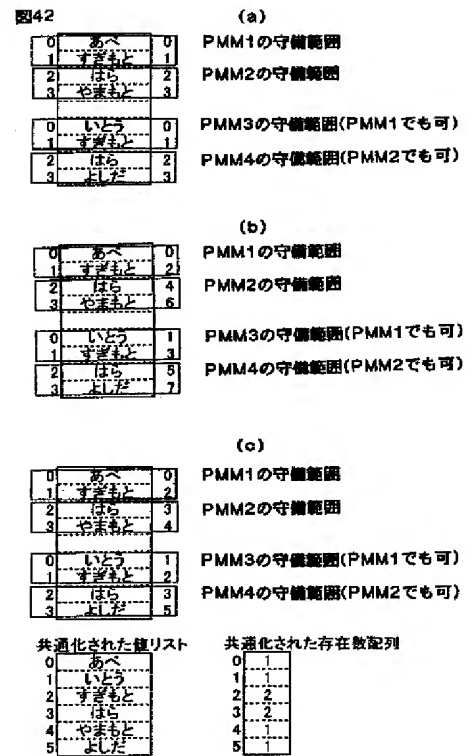
【図31】



【図33】

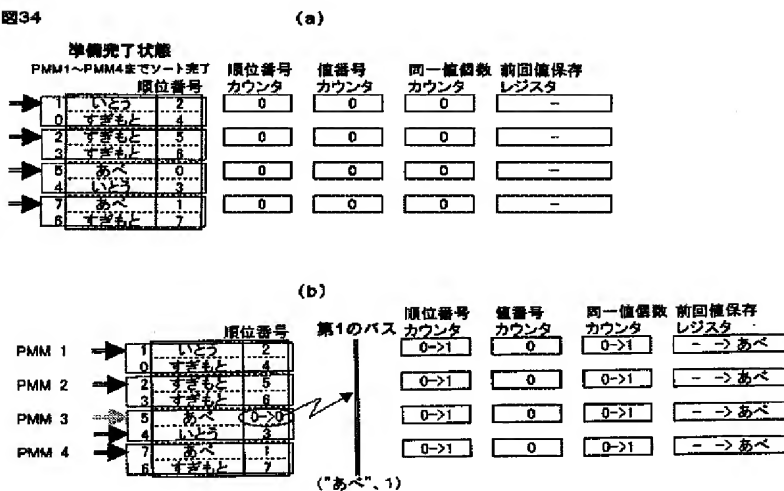


【図42】

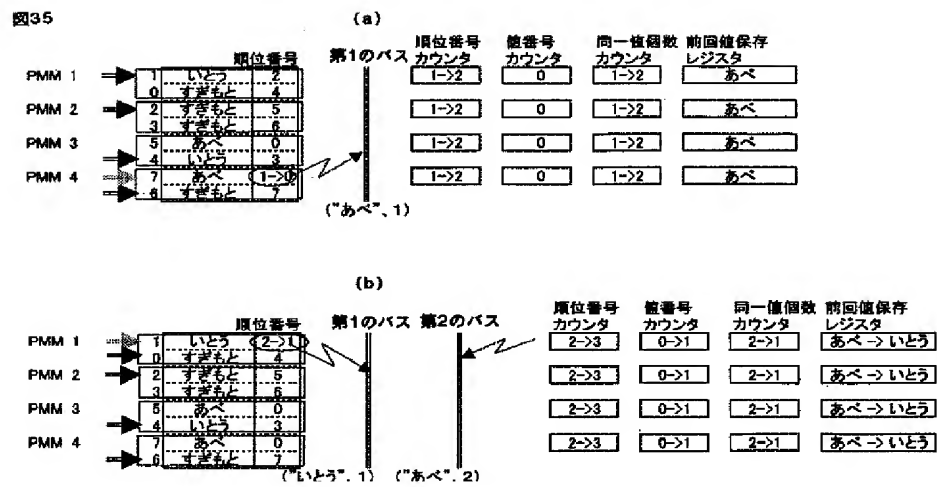


【図34】

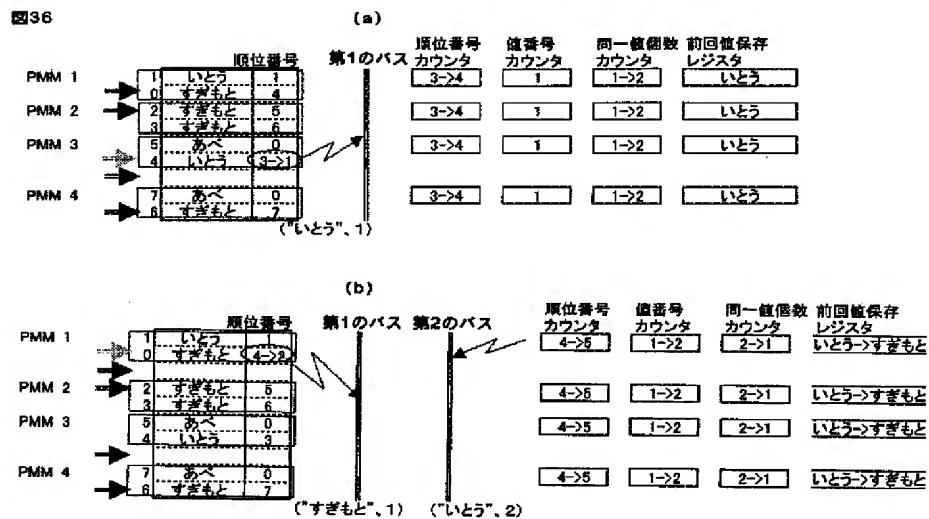
図34



【図35】

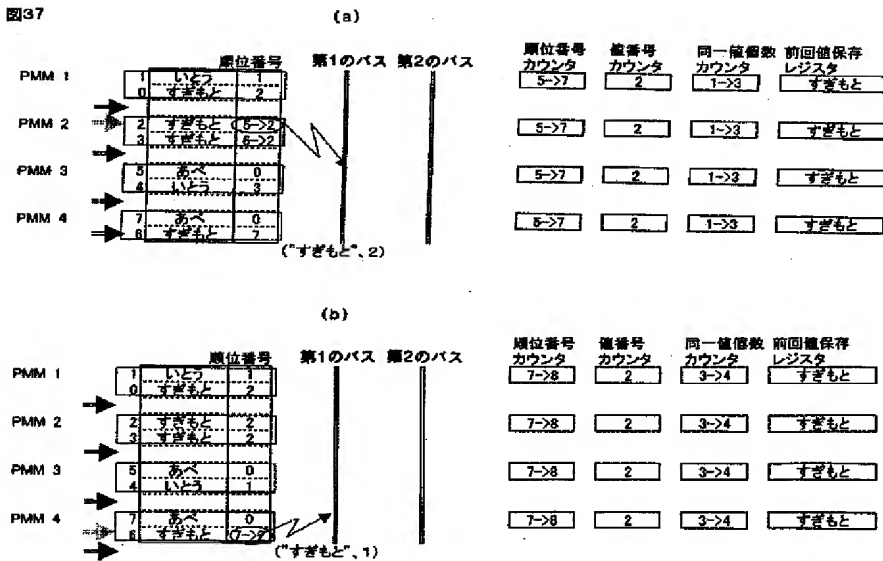


【図36】



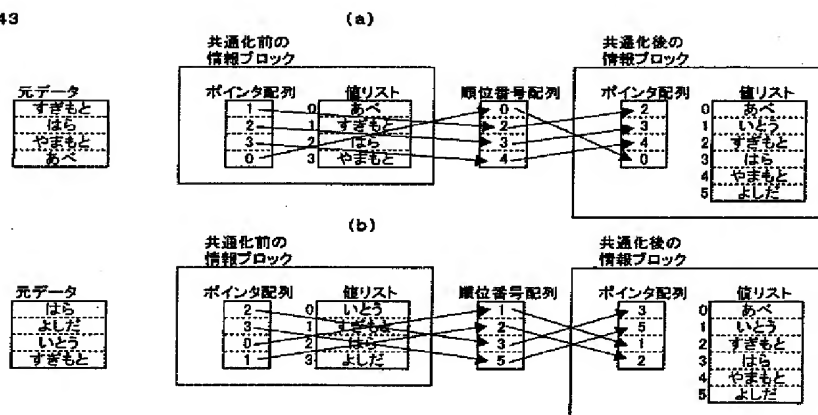
【図37】

図37

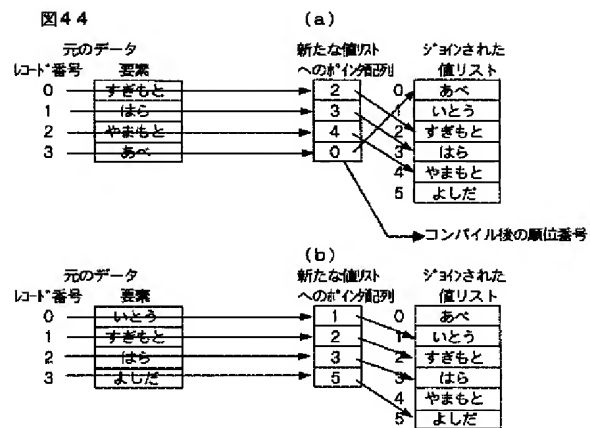


【図43】

図43



【図44】



フロントページの続き

(51)Int.Cl.⁷

識別記号

F I
G 0 6 F 15/403

テーマコード (参考)

3 4 0 D